端到端情感原因对提取的有效子句间模型

摘 要

情感原因对提取旨在从给定文档中提取所有情感子句及其原因子句。先前的工作采用了两步方法，其中第一步分别提取情感子句和原因子句，第二步训练分类器以过滤出否定对。然而，这种用于情感原因对提取的流水线式系统是次优的，因为它存在错误传播的问题，并且两个步骤可能不能很好地相互适应。在本文中，我们从排序的角度来处理情感原因对提取，即对文档中的候选子句对进行排序，并提出了一种一步方法，该方法强调从句间建模以执行端到端提取。它对文档中的子句之间的相互关系进行建模，使用图形注意力学习子句表示，并使用基于内核的相对位置嵌入来增强子句对表示，以实现有效排序。实验结果表明，我们的方法显著优于当前的两步系统，特别是在一个文档中提取多个对的情况下。

1 引言

近年来，情绪原因分析在情绪分析和文本挖掘领域引起了越来越多的研究关注(Lee et al, 2010a;Russo 等人， 2011;nevikaya and Aono, 2013;Ghazi et al, 2015;Gui 等，2016)。它的目标是检测文本中表达某种情绪的原因或刺激因素。理解一种情绪产生的原因具有广泛的应用，如消费者评论挖掘和舆论监测。

以往的研究大多集中在情绪原因提取任务上，该任务旨在识别特定情绪的原因。Xia 和 Ding(2019)指出，这种设置忽略了情绪和原因的相互指示，需要提前对情绪进行注解限制了应用范围。

为了克服这种局限性，他们提出了一项新的研究任务——情感-原因对提取，旨在从给定的文档中提取所有情感表达子句及其原因。如下例所示，一个情感子句 c3 与其对应的原因子句 c2 构造了一个情感-原因对(c3, c2):他告诉我们，自从他生病以来(c1)，他的同学和导师在学业上给了他很多帮助(c2)。他被感动了(c3)，并说他会报答他们(c4)。

与情绪原因提取相比，情绪-原因对提取是一项更具挑战性的任务，因为我们需要全面了解文档的内容和结构来进行情绪-原因共提取，区分情绪-原因子句对和消极子句对。

Xia 和 Ding(2019)提出使用两步解决方案来解决情绪原因对提取问题。第一步，多任务 LSTM 网络分别提取情感从句和原因从句。然后在第二步，使用二进制分类器从所有可能的对中过滤出负对。虽然两步解决方案已经证明了其有效性，但这种流水

线式系统在情感-原因对提取方面并不理想，因为它面临着误差传播的问题，两步之间可能不太适应。

连贯文档有一个基本结构(Mann and Thompson, 1988;Marcu, 2000)，情绪-原因对的两个子句之间存在因果关系，这区别于文献中的其他非情绪-原因对。因此，了解文档中子句之间的相互关系有助于提取潜在的情绪-原因对。

此外，根据语篇的衔接和连贯(De Beaugrande and Dressler, 1981)，两个距离较远的从句包含 3172 个因果关系的概率相对较小。因此，子句对中两个子句之间的相对位置信息可以被认为是提取情感原因对的有效特征。

基于以上两点考虑，本文从排序的角度处理情绪-原因对提取，即对给定文档中的候选子句对进行排序，并提出了一种强调子句间建模的一步法来进行端到端提取。我们的方法首先通过利用图注意来学习子句表示，对子句之间的关系进行建模，通过捕获两个子句之间的潜在关系来促进对提取。

然后它学习子句对的表示，并对这些子句对进行排序，以提取情绪-原因对。提出了一种基于核的相对位置嵌入方案，对相对位置之间的相互影响进行建模，增强子句对表示，实现有效排序。我们将这两个组件集成到一个统一的神经网络中，该网络是端到端的优化。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绪原因对。

这项工作的主要贡献总结如下。

•我们提出了情感原因对提取的第一种端到端方法，这是从排名角度处理这一任务的统一模型。

•我们的方法通过集成子句间关系建模和基于内核的相对位置增强的子句对排序来强调子句间建模。

•实验结果表明，我们的一步方法显著优于当前性能最佳的系统，尤其是在一个文档中提取多个对的情况下。

2 问题定义

最近，由于新发布的带有文本数据的 ERC 数据集引起了自然语言处理(NLP)研究人给定文档 D=（c1，c2，…，c|D|），其中|D|是子句的数量，第 i 个子句 ci=（wi1，wi2，…，wi|ci|）是单词序列，我们的目标是提取 D 中的所有情感原因对：P={（cemo1，ccau1），（cemo2，ccau2），…}，（1），其中（cemoj，ccauj）是第 j 对，cemoj∈D 是情感子句，而 ccauj∈D 是相应的原因子句。注意，一种情绪可能有不止一个原因，同一个原因也可能成为多种情绪的刺激。

3 方法

我们提出了一种名为 RANKCP 的一步方法，该方法对文档中的子句对候选进行排序，以提取情感原因对。总体架构如图 1 所示，由三个组件组成。第一个组件学习给定文档中子句的向量表示。

第二个组件对子句之间的关系进行建模，以获得更好的子句表示。第三部分学习通过相对位置建模增强的子句对表示，并对子句对候选进行排序以提取情感原因对。

3.1 文本编码

给定由|D|子句组成的文档 D=（c1，c2，…，c|D|），我们使用分层递归神经网络（hierarchical RNN）来编码文本内容并学习子句表示。1 对于每个子句 ci=（wi1，wi2，…，wi|ci|），我们使用单词级双向 RNN 来编码其内容信息并获得子句的隐藏状态序列（hi1，hi2，……，hi|ci |）。采用注意层来组合它们，并返回子句 ci 的状态向量 hi，多层感知器（MLP）由 Wa、ba 和 Wa 参数化。然后，文档 D 的子句状态序列（h1，h2，…，h|D|）被馈送到子句级双向 RNN 以产生子句表示，表示为（c1，c2，…，c|D|)。

3.2 用图形注意力网络建立从句间关系

关于从句间关系的知识对于提取情感原因对是有用的。在学习了文档的子句表示之后，为了增强文档中子句之间的交互作用，我们将文档结构视为一个完全连接的子句图，并采用图注意力网络（V eli´ckovi´c 等人，2018）来建模子句间关系。

具体来说，全连通图中的每个节点都是文档中的一个子句，每两个节点都有一条边。我们为每一个节点加了自循环，因为情感子句的原因子句可能是它自己。图注意网络通过堆叠多个图注意层在子句之间传播信息，其中每一层都要通过使用自我注意聚合相邻子句的信息来学习更新的子句表示（V aswani 等人，2017）。

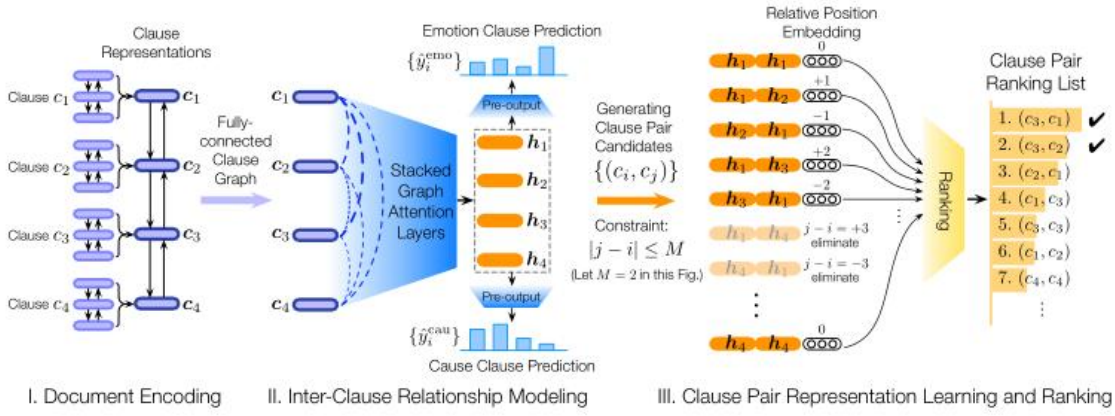

Figure 1: Overview of RANKCP, our proposed one-step approach for emotion-cause pair extraction.

在第 t 个图注意层，设{h（t−1）1，h（t﹣1）2，…，h（t−1）|D|}表示该层的输入子句表示，其中子句 ci 的子句表示表示为 h（t-1）i∈Rdt−1。图形注意机制通过以下聚合方案对文档中的每个子句 ci 进行操作：

$$\boldsymbol{h}_i^{(t)} = \mathrm{ReLU}\left(\sum_{j\in\mathcal{N}(i)} \alpha_{ij}^{(t)} \boldsymbol{W}^{(t)} \boldsymbol{h}_j^{(t-1)} + \boldsymbol{b}^{(t)}\right), \tag{2}$$

其中 h（t）i 是输出表示，W（t）和 b（t）是可学习的参数，N（i）表示 ci 的直接相邻子句（在我们的例子中，它包含文档中的所有子句）。注意力权重 α （t）ij 反映了子句 ci 和子句 cj 之间的聚合水平的强度，这是通过由 w（t）参数化的 MLP 学习的

$$e_{ij}^{(t)} = \boldsymbol{w}^{(t)\top} \tanh\left(\left[\boldsymbol{W}^{(t)}\boldsymbol{h}_i^{(t-1)}; \boldsymbol{W}^{(t)}\boldsymbol{h}_j^{(t-1)}\right]\right),$$
$$\alpha_{ij}^{(t)} = \frac{\exp\left(\mathrm{LeakyReLU}(e_{ij}^{(t)})\right)}{\sum_{k\in\mathcal{N}(i)} \exp\left(\mathrm{LeakyReLU}(e_{ik}^{(t)})\right)}, \tag{3}$$

其中[•； •]是串联。以下矩阵形式可以描述第 t 个图关注层

$$\boldsymbol{H}^{(t)} = \mathrm{ReLU}\left(\boldsymbol{A}^{(t)}\boldsymbol{H}^{(t-1)}\boldsymbol{W}^{(t)\top} + \boldsymbol{b}^{(t)}\right), \tag{4}$$

其中[A（t）]ij= α （t）ij。第一层的输入 H（0）=？c1，c2，c|D|？>是文档编码器的输出（参见第 3.1 节）。通过堆叠 T 个层来建模子句间关系，最后一层的输出是更新的子句表示 H（T）=？h1，h2，h|D|？>。我们进一步采用多头部注意力，其中

每个头部可以基于图注意力的顺序保持特性捕获全局模式（Qiu 等人，2018）。在实践中，我们在每两个相邻层之间添加一条高速公路连接（Srivastava 等人，2015），以控制信息流。

基于使用由多个图注意力层组成的图注意力网络对子句之间的交互进行建模，通过自适应地融合其他子句的信息来生成每个子句表示 hi，并且可以充分学习文档中的子句间关系。

在获得更新的子句表示{hi}|D|i=1 之后，我们将它们输入到两个预输出层中，以预测子句是否为情感/原因子句。具体而言，具有逻辑函数 σ （ • ）的 MLP 用于预测子句 ci 为情感子句的概率：

$$\hat{y}_i^{\text{emo}} = \sigma\left(\boldsymbol{w}_{\text{emo}}^{\top}\boldsymbol{h}_i + b_{\text{emo}}\right). \qquad (5)$$

类似地，另一层获得了子句 ci 为原因子句的概率。

3.3 基于核的相对位置嵌入的子句对排序

我们假设，如果两个从句的相对位置太大，它们形成情感原因对的概率很小。因此，给定文档 D=（c1，…，c|D|），我们考虑其中两个子句的相对位置（绝对值）|j−i|小于或等于某个值 M 的每个子句对（ci，cj）作为情感原因对的候选。我们从文档 D 中构造一组候选子句对：

$$\mathcal{P}' = \{(c_i, c_j) \mid -M \le j - i \le +M\}. \qquad (6)$$

学习子句对表示

对于每个候选子句对 pij=（ci，cj）∈P0，其初始化表示是通过串联三个向量获得的：子句 ci 的表示 hi，子句 cj 的表示 hj，以及它们的相对位置 j−i 的嵌入 rj−i。我们使用一个单层 MLP 来学习其表示：

$$\boldsymbol{p}_{ij} = \mathsf{ReLU}(\boldsymbol{W}_p[\boldsymbol{h}_i; \boldsymbol{h}_j; \boldsymbol{r}_{j-i}] + \boldsymbol{b}_p), \qquad (7)$$

具有可学习的 Wp 和 bp。接下来我们介绍如何构建相对位置嵌入。

原始相对位置嵌入

对于每个相对位置 m∈{−m，…，−1，0，+1，…，+m}，我们通过从均匀分布采样来随机初始化嵌入 rm。然后将每个相对位置嵌入与模型训练过程一起学习。

基于核的相对位置嵌入

除了上述每个相对位置嵌入部分彼此独立的普通方案之外，我们的目标是对不同

相对位置之间的相互影响进行建模，以进一步改进相对位置嵌入。为此，对于每个相对位置 m∈{−m，…，+m}，我们使用 RBF 核函数 Km（ • ）来建模 m 和其他相对位置之间的影响：

$$\mathsf{K}_m(j) = \exp\left(-\frac{(j-m)^2}{{\sigma_\mathsf{K}}^2}\right), \qquad (8)$$

其中 j∈{−M，…，+M}是可能的相对位置值之一， σ K 限制核函数的形状。

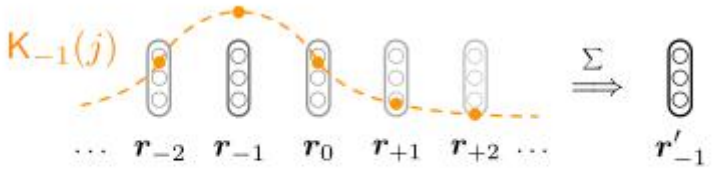

Figure 2: An example: calculating $\boldsymbol{r}'_{-1}$ using kernel.

然后，我们增强原始的嵌入，通过整合其他相对位置的影响来嵌入 rm：

$$\boldsymbol{r}'_m = \sum_{j=-M}^{+M} \mathsf{K}_m(j) \cdot \boldsymbol{r}_j \,. \qquad (9)$$

其背后的直觉是，如果 j 接近 m，rj 将比其他遥远的相对位置对 r0m 施加更大的影响。图 2 显示了 m=−1 的图解。作为 σ K→ 0，基于内核的嵌入会退化为普通嵌入。因此，我们的基于内核的嵌入方案可以被视为原始嵌入的正则化版本。

排序子句对

采用具有激活函数事实（ • ）的排名层（由 wr 和 br 参数化）来产生每个子句对候选 pij∈P0 的排名分数：

$$\hat{y}_{ij} = f_{\text{act}}\left(\boldsymbol{w}_r^\top \boldsymbol{p}_{ij} + b_r\right) . \qquad (10)$$

3.4 优化

我们的网络 RANKCP 经过端到端优化。

输入文档 D 的损失函数由以下两部分组成。

第一部分测量子句对的排名得分。逐点排名损失定义为：

$$\mathcal{L}_{\text{pair}} = \sum_{p_{ij} \in \mathcal{P}'} -(y_{ij} \log \hat{y}_{ij} + (1-y_{ij}) \log(1-\hat{y}_{ij})), \quad (11)$$

其中 yij∈{0,1}是子句对 pij 的基本真值（yij=1 表示 pij 是情绪原因对），事实（•）被设置为逻辑函数。

它也可以通过成对的排序损失来计算，具有裕度超参数 γ：

$$\mathcal{L}_{\text{pair}} = \sum_{\{p^+, p^-\} \in \mathcal{P}'_{p^+ \succ p^-}} \max\{0, -(y^+ - y^-) + \gamma\}, \quad (12)$$

其中，子句对 p+的基真值为 1，而子句对 p−的基真为 0（因此 p+的分数 y+应该高于 p−的分数 y−），并且事实（•）被设置为 tanh 函数。

根据子句对的基本真理，我们知道子句是否是情绪/原因子句，因此我们使用两个交叉熵损失函数 Lemo 和 Lcau 来监督两个预输出预测。

我们采用上述两部分的总和作为文件 D 的最终损失函数 L：

$$\mathcal{L} = \mathcal{L}_{\text{pair}} + (\mathcal{L}_{\text{emo}} + \mathcal{L}_{\text{cau}}). \quad (13)$$

这就形成了对子句表示学习和子句对排序的两级监督。

3.5 基于词典的提取

在测试时，一个关键问题是如何根据所有候选配对的排名分数来提取潜在的情感原因配对。请注意，要确定可用于所有文档的总体阈值分数，以将考生分为情绪原因对和负面原因对，并不容易。

我们采用基于词典的提取方案来从测试文档的前 N 个排名列表{p1，p2，…，pN}中获得情感原因对。我们首先提取最高的对 p1（得分最高）作为情感原因对。然后，对于每个剩余的子句对 pi=（ci，1，ci，2）∈{p2，…，pN}，我们使用情感词典来确定子句 ci，1 是否包含情感词。如果是这样，我们将 pi 对提取为情感原因对。因此，我们的模型能够从给定文档中提取多个情感原因对。

4 实验

我们进行了广泛的实验来验证我们提出的模型 RANKCP 的有效性。

4.1 实验设置

我们使用（Xia 和 Ding，2019）发布的基准数据集进行实验。该数据集基于情感原因提取语料库（Gui 等人，2016）构建，该语料库由来自 SINA NEWS 网站的 1945 篇中文文档组成。表 1 显示了汇总统计数据。在我们的实验中，遵循之前的工作，我们使用相同的数据分割（10 倍交叉验证），并选择精度 P、召回率 R 和 F 分数 F1 作为评估指标：

$$
\begin{aligned}
P &= \frac{\#\text{correctly predicted pairs}}{\#\text{predicted pairs}}, \\
R &= \frac{\#\text{correctly predicted pairs}}{\#\text{ground-truth pairs}}, \\
F_1 &= \frac{2 \cdot P \cdot R}{P + R}.
\end{aligned} \tag{14}
$$

此外，我们还分别评估了情感子句提取和原因子句提取的性能。也就是说，我们将情感原因对分解为一组情感子句和一组原因子句，然后计算这两组原因的度量。准确度、回忆和 F 分数的定义与等式 14 中的定义相似：用“情感子句”或“原因子句”替换“成对”。

Xia 和 Ding（2019）提出了三个两步系统。第一步分别提取情感子句和原因子句，第二步是过滤出否定对的二元分类器。

具体来说，他们三个系统的差异存在于第一步。

•INDEP 使用双向 LSTM 对子句进行编码，然后使用两个独立的双向 LSTM 分别提取情感和原因子句。

•INTER-CE 与 INDEP 的不同之处在于，它首先提取原因子句，然后将预测的分布用作提取情感子句的额外特征。

•INTER-EC 与 INTER-CE 相似，只是它首先提取情感子句。

为了公平比较，我们采用了 INTER-EC 中使用的相同单词嵌入。我们使用 LSTM 作为 RNN 单元，子句表示的维数为 200。我们堆叠了两个图形注意力层来构建图形注意力网络，并为每个层添加了 0.1 的丢弃率。最大相对位置 M 设置为 12，相对位置嵌

入维数设置为 50，RBF 核函数中 σ K=1

我们使用 Adam 优化器以 0.001 的学习率和 4 个小批量大小训练 RANKCP，'2 正则化系数设置为 1e-5。我们选择逐点排名损失，因为使用它的训练比使用成对损失的训练更快。我们使用 ANTUSD（Wang 和 Ku，2016）作为情绪词典，4，超参数 N 设置为 3。

4.2 实验结果

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**

Table 2: Experimental results on emotion-cause pair extraction. Moreover, results on emotion clause extraction and cause clause extraction are also reported. "RANKCP (top-1)" denotes the model that does not use the lexicon-based extraction at test time, and directly chooses the clause pair having the highest ranking score as the unique emotion-cause pair for a document.

表 2 给出了情绪原因对提取和两个子任务情感子句提取和原因子句提取的对比结果。5 我们的一步方法 RANKCP 在所有三个任务上都比其他基线系统显示出明显的优势，在三个任务上分别比表现最好的基线系统 INTER-EC 获得 4.82%，3.18%和 3.17%的 F1 改进。

更具体地说，我们可以观察到，上述优势主要来自于召回率 r 的显著提高，与 INTEREC 相比，RANKCP 在情绪-原因对提取和原因从句提取上分别提高了 8.43%和 6.60%，这表明我们的一步解决方案可以在不影响精度 P 的情况下有效地提取更多正确的情绪-原因对。

表 2 中最后两行结果的比较说明了基于词汇提取的有效性。我们可以看到，添加基于词汇的提取方案可以提高回忆率 R，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，但 F-score F1 仍然比仅提取文档中的前 1 对表现更好。因此，基于词典的提取是一种有效的方法。

# 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**

Table 3: Comparative results for documents with only one and more than one emotion-cause pair.

我们进一步比较了在一个文档中提取多个对的结果。我们将每个折叠的测试集分为两个子集:一个子集包含只有一个情绪-原因对的文档，另一个子集包含有两个或更多情绪-原因对的文档。

表 3 分别报告了两个子集的比较结果。可以看到，我们的模型在两个子集上都优于 INTER-EC。我们的一步法对于具有多个情绪原因对的文档相对更有效(F1 改善超过 13%)

我们还提供了与最近提出的情绪原因提取任务方法的比较结果:基于规则的方法 RB (Lee 等人，2010a)，基于传统机器学习的方法 MULTI-KERNEL (Gui 等人，2016)，以及三种神经方法 CONVMS-MEMNET (Gui 等人，2017)，CANN (Li 等人，2018)和 RTHN (Xia 等人，2019)。注意，它们都使用已知的情感子句作为模型输入。表 4 的上半部分报告了它们的性能。

表 4 的下半部分显示了不使用已知情感从句作为模型输入的方法的比较结果。结果表明我们提出的 RANKCP 方法比其他方法的性能要好得多。此外，RANKCP 虽然没有利用测试文档的已知情绪作为模型输入，但仍然优于 RB 和 MULTI-KERNEL，与 CONVMSMEMNET 相当。因此，我们的方法受益于子句间建模，并显示了其在原因子句提取上的有效性。

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

Table 4: Results on emotion cause extraction task. CANN – E and RTHN-APE denote the variant models of CANN and RTHN respectively, which do not utilize known emotion clauses as model input.

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**

Table 5: Comparison of different supervised signals for RANKCP.

4.3 进一步的讨论

两级监管的效果

我们的模型是用两个监督信号的混合来训练的:在图注意网络输出的子句表示学习上的低级信号 Lemo + Lcau(见公式 5)，以及在子句对表示学习和排序上的高级信号 Lpair(见公式 10)。为了验证低层次监督的效果，我们只使用 Lpair 训练我们的模型，与完整模型的比较结果如表 5 所示。结果表明，两级监督训练提高了提取效果。这表明引入低级监督有助于学习更好的子句表示，并最终促进子句对表示的学习和排序过程。

图注意层的效果

用于建模子句间潜在关系的图注意网络是我们方法的关键组成部分。我们改变图注意层数(从 0 到 3)来测试其效果，情绪-原因对提取和原因子句提取的结果如图 3 所示。

显然，没有图注意层的模型无法获得良好的性能。我们的方法在两层图注意网络中获得了最佳性能，这表明在这项任务中可以充分建模子句间关系，而无需堆叠大量层。

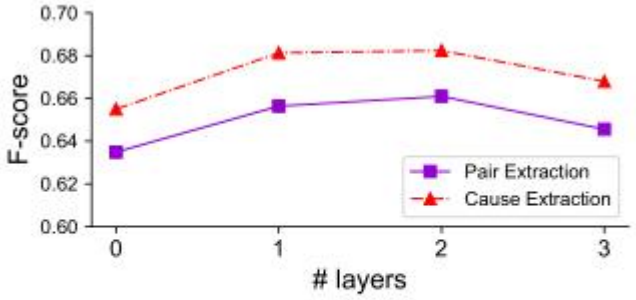

Figure 3: Results of RANKCP with various graph attention layers.

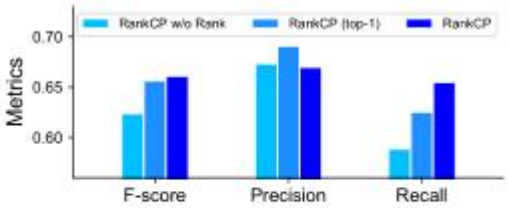

Figure 4: Comparative results of our variant model that removes the clause pair representation learning and ranking component (denoted as "RANKCP w/o Rank") and our full model RANKCP.

子句对表示学习的效果

我们进一步研究了直接使用从句表征来预测情感从句和原因从句是否能获得理想的表现。换句话说，我们去除子句对表示学习和排名组件，并利用图注意网络的预测(即公式 5)来产生情绪-原因对。在对文档中的情绪从句和原因从句进行预测后，我们将预测的情绪和原因的所有组合视为提取的情绪-原因对，该变体模型与我们的完整

模型的对比结果如图 4 所示。

相对位置嵌入的效果

RANKCP 的性能比变体要好得多(尤其是在召回方面)，说明只提供子句级预测不适合情感-原因对提取任务。因此，在统一的一步模型中结合子句级和子句对表示学习对于提取情绪-原因对确实是有效的。

我们去掉 RANKCP 中的相对位置嵌入部分来验证其效果。我们还比较了原始和基于内核的相对位置嵌入方案。结果如表 6 所示。

去除相对位置嵌入会导致性能下降，这表明子句对之间的相对位置确实对预测有用。从前两行还可以观察到，基于词库的提取并不能优于 top-1 提取，这进一步验证了没有相对位置嵌入的模型不能提供理想的排名表。基于内核的嵌入在 top-1 和基于 lexicon 的提取上都比原始嵌入表现得更好，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，进一步提高情感-原因对提取的性能。

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**

Table 6: Comparison on relative position embedding schemes. "ext." is the abbreviation for "extraction".

4.4 案例分析

我们展示了一个文档，我们的方法 RANKCP 正确地提取了它的情绪-原因对(c5, c4)，而 INTER-EC 失败:

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)。

我们将图 5 中两个从句 c4 和 c5 的注意力权重可视化。情感子句 c5 以最高权重参与相应的原因 c4 ，说明图注意有效地捕捉了两个子句之间的关系。

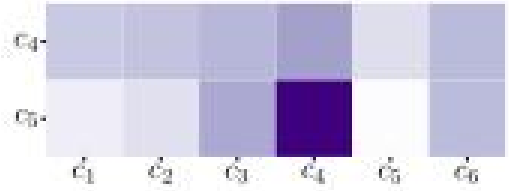

Figure 5: Attention weights for two clauses c_4 and c_5.

5 相关工作

情感原因提取

Lee 等人(2010a,b)首先研究了情绪原因提取，并设计了一个基于语言规则的系统来检测原因事件。早期工作尝试基于规则(Chen 等人，2010;nevikaya and Aono, 2013;Gao 等人，2015)，基于常识(Russo 等人，2011)，以及基于传统机器学习(Ghazi 等人，2015)的方法来提取某些情绪表达的原因。

Gui 等人(2016)提出了事件驱动的多核 SVM 方法，并发布了基准语料库。基于特征的方法(Xu 等人，2019)和神经方法(Gui 等人，2017;李等，2018;丁等，2019;Y u et al, 2019)最近被提出。Xia 等人(2019)采用了增加位置信息和集成全局预测嵌入的变压器编码器来提高性能。Fan 等人(2019)采用情绪和位置正则器来抑制参数学习。Hu 等(2019)利用外部情感分类语料库对模型进行预训练。

在其他研究方向上，一些工作(Cheng et al, 2017)提取了多用户结构微博语境下的情感原因。此外，Kim and Klinger(2018)和 Bostan et al(2020)将情绪视为结构化现象，并研究了情绪的语义作用，包括触发短语、体验者、目标和原因以及读者的感知。

情感-原因对提取

以往的情绪原因分析研究都需要以已知的情绪子句作为模型输入。开创性工作(Xia and Ding, 2019)首次提出情绪-原因对提取任务。他们提出了一种两步走的方法，分别提取情感子句和原因子句，然后训练分类器过滤掉负面子句。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实现端到端情绪原因对提取的一步解决方案，实现了显著更好的性能。

6 总结与展望

在本文中，我们提出了第一个一步神经方法 RANKCP 来解决情绪-原因对提取问题，该方法强调从排名的角度进行子句间建模。

我们的方法有效地建模子句之间的关系来学习子句表示，并将相对位置增强的子句对排名集成到统一的神经网络中，以端到端方式提取情绪原因对。在基准数据集上的实验结果表明，RANKCP 的性能显著优于以前的系统，进一步的分析验证了我们模型中每个组件的有效性。

在今后的工作中，我们将在以下几个方面进行探索。首先，目前对情绪原因分析的研究主要集中在子句级的粗粒度提取上，需要进一步设计细粒度提取跨级或短语级情绪表达和原因的方法。其次，设计有效的方法将适当的语言知识注入神经模型，这

对情感分析任务是有价值的(Ke 等人，2019;Zhong 等，2019)。最后，研究情绪的语义角色会很有趣(Bostan 等人，2020 年)，它考虑了情绪表达的完整结构，更具挑战性。