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

Penghui Wei, Jiahao Zhao, Wenji Mao
[†]SKL-MCCS, Institute of Automation, Chinese Academy of Sciences (CASIA)
[‡]School of Artificial Intelligence, University of Chinese Academy of Sciences
{weipenghui2016,zhaojiahao2019,wenji.mao}@ia.ac.cn

Abstract

Emotion-cause pair extraction aims to extract all emotion clauses coupled with their cause clauses from a given document. Previous work employs two-step approaches, in which the first step extracts emotion clauses and cause clauses separately, and the second step trains a classifier to filter out negative pairs. However, such pipeline-style system for emotion-cause pair extraction is suboptimal because it suffers from error propagation and the two steps may not adapt to each other well. In this paper, we tackle emotion-cause pair extraction from a ranking perspective, i.e., ranking clause pair candidates in a document, and propose a one-step neural approach which emphasizes inter-clause modeling to perform end-to-end extraction. It models the interrelations between the clauses in a document to learn clause representations with graph attention, and enhances clause pair representations with kernel-based relative position embedding for effective ranking. Experimental results show that our approach significantly outperforms the current two-step systems, especially in the condition of extracting multiple pairs in one document.

1 Introduction

Emotion cause analysis has attracted increasing research attention in sentiment analysis and text mining community in recent years (Lee et al., 2010a; Russo et al., 2011; Neviarouskaya and Aono, 2013; Ghazi et al., 2015; Gui et al., 2016). Its goal is to detect causes or stimuli for a certain emotion expressed in text. Understanding why an emotion occurs has broad applications such as consumer review mining and public opinion monitoring.

Previous studies mostly focus on *emotion cause extraction* task which aims to identify cause(s) for a *given* emotion. Xia and Ding (2019) pointed out that this setting ignores the mutual indication of emotions and causes, and the need of emotion annotation in advance restricts the range of applications. To overcome such limitations, they put forward a new research task named *emotion-cause pair extraction*, aiming to extract all emotion expression clauses coupled with their causes from a given document. As shown in the following example, an *emotion clause* c_3 and its corresponding *cause clause* c_2 construct an *emotion-cause pair* (c_3, c_2):

Example. *He told us that since his illness* (c_1), *his classmates and advisors have given him much help about the schoolwork* (c_2). *He has been* ***touched*** (c_3), *and said that he will repay them* (c_4).

Compared with emotion cause extraction, emotion-cause pair extraction is a more challenging task, because we need a comprehensive understanding of document content and structure to perform emotion-cause co-extraction and discriminate emotion-cause clause pairs from negative ones.

Xia and Ding (2019) proposed to tackle emotion-cause pair extraction using a two-step solution. At the first step, a multi-task LSTM network extracts emotion clauses and cause clauses separately. Then at the second step, a binary classifier is used to filter out negative pairs from all possible pairs. Although the two-step solution has shown its effectiveness, such pipeline-style system is suboptimal for emotion-cause pair extraction, because it is confronted with error propagation, and the two steps may not adapt to each other well.

Coherent document has an underlying structure (Mann and Thompson, 1988; Marcu, 2000) and there is a causal relationship between the two clauses of an emotion-cause pair, which distinguishes it from other non-emotion-cause pairs in the document. Thus, knowledge about the interrelations between the clauses in a document is beneficial for extracting potential emotion-cause pairs. Further, according to the cohesion and coherence of discourse (De Beaugrande and Dressler, 1981), the probability of two distant clauses containing

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 3171–3181
July 5 - 10, 2020.

causal relationship is relatively small. Thus, relative position information between two clauses of a clause pair can be considered as an effective feature for emotion-cause pair extraction.

Based on the above two considerations, in this paper, we tackle emotion-cause pair extraction from a ranking perspective, i.e., ranking clause pair candidates in a given document, and propose a one-step approach which emphasizes inter-clause modeling to perform end-to-end extraction. Our approach first models the inter-clause relationships via exploiting graph attention to learn clause representations, facilitating pair extraction through capturing the latent relationship between two clauses. It then learns clause pair representations and rank these pairs to extract emotion-cause pairs. A kernel-based relative position embedding scheme is proposed to model the mutual impact among relative positions and enhance clause pair representations for effective ranking. We integrate the two components into a unified neural network, which is optimized end-to-end. Unlike the previous two-step solution, our approach can directly extract emotion-cause pairs from documents.

The main contributions of this work are summarized as follows.

- To our knowledge, we propose the first end-to-end approach for emotion-cause pair extraction, which is a unified model to tackle this task from a ranking perspective.
- Our approach emphasizes inter-clause modeling by integrating inter-clause relationship modeling and kernel-based relative position enhanced clause pair ranking.
- Experimental results demonstrate that our one-step approach significantly outperforms the current best-performing systems, especially in the condition of extracting multiple pairs in one document.

2 Problem Formulation

Given a document $D = (c_1, c_2, \ldots, c_{|D|})$ where $|D|$ is the number of clauses and the i-th clause $c_i = (w_1^i, w_2^i, \ldots, w_{|c_i|}^i)$ is a word sequence, our goal is to extract all emotion-cause pairs in D:

$$\mathcal{P} = \{(c^{\text{emo}_1}, c^{\text{cau}_1}), (c^{\text{emo}_2}, c^{\text{cau}_2}), \ldots\}, \quad (1)$$

where $(c^{\text{emo}_j}, c^{\text{cau}_j})$ is the j-th pair, $c^{\text{emo}_j} \in D$ is an emotion clause, and $c^{\text{cau}_j} \in D$ is the corresponding cause clause. Note that an emotion may have more than one cause, and the same cause may also become the stimulus of multiple emotions.

3 Proposed Approach

We propose a one-step approach named RANKCP, which ranks clause pair candidates in a document to extract emotion-cause pairs. The overall architecture is shown in Fig. 1, which consists of three components. The first component learns vector representations of clauses in a given document. The second component models the relationships between clauses to obtain better clause representations. The third component learns clause pair representations enhanced with relative position modeling, and ranks clause pair candidates to extract emotion-cause pairs.

3.1 Document Encoding

Given a document $D = (c_1, c_2, \ldots, c_{|D|})$ composed of $|D|$ clauses, we use a hierarchical recurrent neural network (Hierarchical RNN) to encode textual content and learn clause representations.[1]

For each clause $c_i = (w_1^i, w_2^i, \ldots, w_{|c_i|}^i)$, we use a word-level bidirectional RNN to encode its content information and obtain the clause's hidden state sequence $(\boldsymbol{h}_1^i, \boldsymbol{h}_2^i, \ldots, \boldsymbol{h}_{|c_i|}^i)$. An attention layer is adopted to combine them and return a state vector $\boldsymbol{h}^i = \sum_{j=1}^{|c_i|} \alpha_j \boldsymbol{h}_j^i$ for the clause c_i, where $\alpha_j = \mathsf{Softmax}\left(\boldsymbol{w}_a^\top \tanh(\boldsymbol{W}_a \boldsymbol{h}_j^i + \boldsymbol{b}_a)\right)$ is the attention weight of the j-th word in clause c_i, with a multilayer perceptron (MLP) parameterized by $\boldsymbol{W}_a, \boldsymbol{b}_a$ and $\boldsymbol{w}_a$. Then the document D's clause state sequence $(\boldsymbol{h}^1, \boldsymbol{h}^2, \ldots, \boldsymbol{h}^{|D|})$ is fed into a clause-level bidirectional RNN to produce clause representations, denoted as $(\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_{|D|})$.

3.2 Modeling Inter-Clause Relationships with Graph Attention Network

Knowledge about inter-clause relationships is useful for extracting emotion-cause pairs. After learning clause representations of a document, to enhance the interactions between clauses in the document, we regard the document structure as a fully-connected clause graph, and adopt graph attention network (Veličković et al., 2018) to model the inter-clause relationships.

Specifically, each node in the fully-connected graph is a clause in the document, and every two nodes have an edge. We also add a self-loop edge

[1] Pretrained BERT encoder (Devlin et al., 2019) based clause representation component is shown in Appendix A.1.

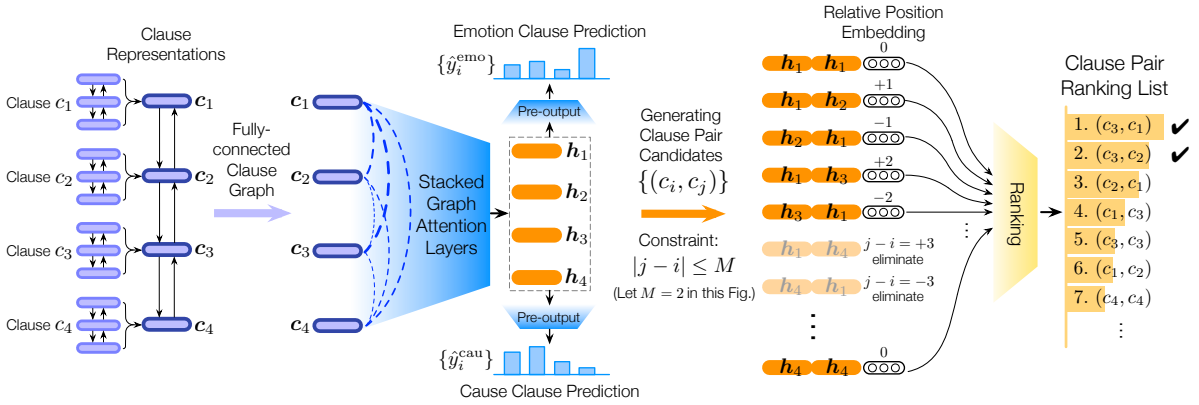

Figure 1: Overview of RANKCP, our proposed one-step approach for emotion-cause pair extraction.

to every node, because the cause clause of an emotion clause may be itself. Graph attention network propagates information among clauses by stacking multiple graph attention layers, in which each layer is to learn an updated clause representation via aggregating neighboring clauses' information using self-attention (Vaswani et al., 2017).

At the t-th graph attention layer, let $\{\boldsymbol{h}_1^{(t-1)}, \boldsymbol{h}_2^{(t-1)}, \ldots, \boldsymbol{h}_{|D|}^{(t-1)}\}$ denote the input clause representations of this layer, where the clause representation of clause c_i is denoted as $\boldsymbol{h}_i^{(t-1)} \in \mathbb{R}^{d_{t-1}}$. The graph attention mechanism operates on each clause c_i in the document via the following aggregation scheme:

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

where $\boldsymbol{h}_i^{(t)}$ is the output representation, $\boldsymbol{W}^{(t)}$ and $\boldsymbol{b}^{(t)}$ are learnable parameters, and $\mathcal{N}(i)$ denotes the directly neighboring clauses of c_i (in our case it contains all clauses in the document). The attention weight $\alpha_{ij}^{(t)}$ reflects the strength of aggregation level between the clause c_i and the clause c_j, which is learned by an MLP parameterized by $\boldsymbol{w}^{(t)}$:

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

$$\alpha_{ij}^{(t)} = \frac{\exp\left(\mathrm{LeakyReLU}(e_{ij}^{(t)})\right)}{\sum_{k \in \mathcal{N}(i)} \exp\left(\mathrm{LeakyReLU}(e_{ik}^{(t)})\right)}, \tag{3}$$

where $[\cdot;\cdot]$ is concatenation. The following matrix form can describe the t-th graph attention layer:

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

where $[\boldsymbol{A}^{(t)}]_{ij} = \alpha_{ij}^{(t)}$. The first layer's input $\boldsymbol{H}^{(0)} = \left(\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_{|D|}\right)^{\top}$ is the document encoder's output (see Section 3.1). By stacking T layers to model inter-clause relationships, the last layer's output is the updated clause representations $\boldsymbol{H}^{(T)} = \left(\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_{|D|}\right)^{\top}$. We further adopt multi-head attention, where each head can capture a global pattern based on the order-preserving property of graph attention (Qiu et al., 2018). In practice, we add a highway connection (Srivastava et al., 2015) between every two adjacent layers to control the information flow.[2]

Based on modeling the interactions between clauses with graph attention network composed of multiple graph attention layers, each clause representation $\boldsymbol{h}_i$ is produced by fusing other clauses' information adaptively, and the inter-clause relationships in the document can be learned sufficiently.

After obtaining updated clause representations $\{\boldsymbol{h}_i\}_{i=1}^{|D|}$, we feed them into two pre-output layers to predict whether a clause is an emotion/cause clause or not. Specifically, an MLP (parameterized by $\boldsymbol{w}_{\mathrm{emo}}$ and b_{emo}) with logistic function $\sigma(\cdot)$ is used to predict the probability of a clause c_i being an emotion clause (denoted as $\hat{y}_i^{\mathrm{emo}}$):

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

Similarly, the probability of a clause c_i being a cause clause ($\hat{y}_i^{\mathrm{cau}}$) is obtained by the other layer.

3.3 Clause Pair Ranking with Kernel-based Relative Position Embedding

To extract emotion-cause pairs in an end-to-end fashion, our approach further learns clause pair rep-

[2] We attempted to extend graph attention to a structured version, but it did not lead to improvement (see Appendix B).

resentations and rank these pairs to obtain emotion-cause pairs. Relative position between two clauses is the key to indicate emotion-cause pairs. Thus, we inject relative position information into the clause pair representation learning process via relative position embedding learning.

We hypothesize that if the relative position of two clauses is too large, the probability of their forming an emotion-cause pair is very small. Thus, given the document $D = (c_1, \ldots, c_{|D|})$, we consider each clause pair (c_i, c_j) in which the two clauses' relative position (absolute value) $|j - i|$ is less than or equal to a certain value M as a candidate of emotion-cause pair. We construct a set of clause pair candidates from the document D:

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

Learning Clause Pair Representations

For each clause pair candidate $p_{ij} = (c_i, c_j) \in \mathcal{P}'$, its initialized representation is obtained by concatenating three vectors: the clause c_i's representation $\boldsymbol{h}_i$, the clause c_j's representation $\boldsymbol{h}_j$, and their relative position $j - i$'s embedding $\boldsymbol{r}_{j-i}$. We employ a one-layer MLP to learn its representation:

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

with learnable $\boldsymbol{W}_p$ and $\boldsymbol{b}_p$. Next we introduce how to build relative position embeddings.

Vanilla relative position embedding For each relative position $m \in \{-M, \ldots, -1, 0, +1, \ldots, +M\}$, we randomly initialize the embedding $\boldsymbol{r}_m$ via sampling from a uniform distribution. Then each relative position embedding is learned together with the model training process.

Kernel-based relative position embedding Beyond the above vanilla scheme where each relative position embedding is partly independent of each other, we aim to model the mutual impact among different relative positions for further improving relative position embeddings. To this end, for each relative position $m \in \{-M, \ldots, +M\}$, we use an RBF kernel function $\mathsf{K}_m(\cdot)$ to model the impact between m and other relative positions:

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

where $j \in \{-M, \ldots, +M\}$ is one of possible relative position values, and σ_K restricts the shape of the kernel function. Then, we enhance the vanilla embedding $\boldsymbol{r}_m$ by integrating other relative positions' influences:

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

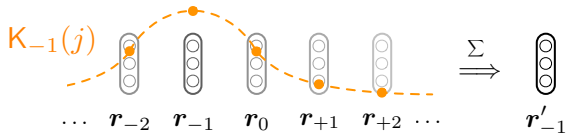

Figure 2: An example: calculating $\boldsymbol{r}'_{-1}$ using kernel.

The intuition behind it is that if j is close to m, $\boldsymbol{r}_j$ will exert more influence on $\boldsymbol{r}'_m$ than other distant relative positions. Fig. 2 shows an illustration for $m = -1$. As $\sigma_\mathsf{K} \to 0$, kernel-based embeddings devolve to vanilla ones. Thus, our kernel-based embedding scheme can be regarded as a regularized version of vanilla embedding.

Ranking Clause Pairs

A ranking layer (parameterized by $\boldsymbol{w}_r$ and b_r) with activation function $f_{\text{act}}(\cdot)$ is adopted to produce the ranking score $\hat{y}_{ij}$ for each clause pair candidate $p_{ij} \in \mathcal{P}'$:

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

3.4 Optimization

Our network RANKCP is optimized end-to-end. The loss function for the input document D consists of the following two parts.

The first part measures the ranking scores of clause pairs. Pointwise ranking loss is defined as:

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

where $y_{ij} \in \{0, 1\}$ is the ground-truth of the clause pair p_{ij} ($y_{ij} = 1$ means that p_{ij} is an emotion-cause pair), and $f_{\text{act}}(\cdot)$ is set to logistic function. It can also be computed by pairwise ranking loss, with a margin hyperparameter γ:

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

where the ground-truth of clause pair p^+ is 1 while the ground-truth of clause pair p^- is 0 (thus p^+'s score y^+ should rank higher than p^-'s score y^-), and $f_{\text{act}}(\cdot)$ is set to $\tanh$ function.

The second part of the loss function measures the pre-output $\hat{y}_i^{\text{emo}}$ and $\hat{y}_i^{\text{cau}}$ of graph attention

# 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

Table 1: Summary statistics of the dataset for evaluation. "Doc." is the abbreviation for "Document".

network (see Eq. 5). According to the ground-truth of clause pairs, we know whether a clause is an emotion/cause clause or not, thus we use two cross-entropy loss functions $\mathcal{L}_{\text{emo}}$ and $\mathcal{L}_{\text{cau}}$ to supervise the two pre-output predictions.

We employ the sum of the above two parts as the final loss function $\mathcal{L}$ for the document D:

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

This forms two-level supervision for both clause representation learning and clause pair ranking.

3.5 Lexicon-based Extraction

At test time, a key problem is how to extract potential emotion-cause pairs according to the ranking scores of all pair candidates. Note that it is not easy to determine an overall threshold score that can be adopted to all documents for dividing candidates into emotion-cause pairs and negative ones.

We adopt a lexicon-based extraction scheme to obtain emotion-cause pairs from the top-N ranking list $\{p^1, p^2, \ldots, p^N\}$ of a test document. We first extract the top pair p^1 (with the highest score) as an emotion-cause pair. Then, for each remaining clause pair $p^i = (c^{i,1}, c^{i,2}) \in \{p^2, \ldots, p^N\}$, we use a sentiment lexicon to determine whether the clause $c^{i,1}$ contains sentiment word(s). If so, we extract the pair p^i as an emotion-cause pair. Therefore, our model is able to extract multiple emotion-cause pairs from a given document.

4 Experiments

We conduct extensive experiments to verify the effectiveness of our proposed model RANKCP.

4.1 Experimental Setup

Dataset and Evaluation Metrics

We use the benchmark dataset released by (Xia and Ding, 2019) to conduct our experiments. This dataset is constructed based on an emotion cause extraction corpus (Gui et al., 2016) that consists of 1,945 Chinese documents from SINA NEWS website. Table 1 shows the summary statistics. In our experiments, following the previous work, we use the same data split (10-fold cross-validation), and choose precision P, recall R and F-score F_1 as evaluation metrics:

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

Moreover, we also evaluate the performance on emotion clause extraction and cause clause extraction respectively. That is, we break down the emotion-cause pairs to a set of emotion clauses and a set of cause clauses, and then compute metrics for the two sets. Precision, recall and F-score are defined similar to those in Eq. 14: replacing "pairs" with "emotion clauses" or "cause clauses".

Comparative Approaches

Xia and Ding (2019) proposed three two-step systems. The first step extracts emotion clauses and cause clauses separately, and the second step is a binary classifier that filters out negative pairs. Specifically, the difference of their three systems exists at the first step.

- INDEP encodes clauses with bidirectional LSTM, then uses two independently bidirectional LSTMs to extract emotion and cause clauses respectively.
- INTER-CE is different from INDEP in that it first extracts cause clauses, and then the predicted distribution is utilized as extra feature to extract emotion clauses.
- INTER-EC is similar to INTER-CE except that it first extracts emotion clauses.

Implementation Details[3]

For fair comparison, we adopt the same word embeddings as used in INTER-EC. We use LSTM as the RNN cell, and the dimension of clause representations is 200. We stack two graph attention layers to build the graph attention network, and we add dropout with rate 0.1 for each layer. The maximum relative position M is set to 12, and the dimension of relative position embedding is set to 50, with $\sigma_{\text{K}} = 1$ in the RBF kernel function.

[3]Our implementation based on `PyTorch` is available at: https://github.com/Determined22/Rank-Emotion-Cause.

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.

We train RANKCP using Adam optimizer with 0.001 learning rate and 4 mini-batch size, and ℓ_2 regularization coefficient is set to 1e-5. We choose pointwise ranking loss because training with it is faster than that with pairwise loss. We use ANTUSD (Wang and Ku, 2016) as the sentiment lexicon,[4] and the hyperparameter N is set to 3.

4.2 Experimental Results

Results on Emotion-Cause Pair Extraction

Table 2 reports the comparative results on emotion-cause pair extraction and two sub-tasks, i.e., emotion clause extraction and cause clause extraction.[5] Our one-step approach RANKCP shows clear advantage over other baseline systems on all three tasks, which obtains 4.82%, 3.18% and 3.17% F_1 improvements over the best-performing baseline system INTER-EC on three tasks respectively.

More specifically, we can observe that the above advantage mainly originates from the significant improvement of recall R. Comparing to INTER-EC, RANKCP achieves 8.43% and 6.60% improvements on emotion-cause pair extraction and cause clause extraction respectively, which indicates that our one-step solution can effectively extract more correct emotion-cause pairs without hurting the precision P.

Comparison between the last two lines' results in Table 2 demonstrates the effectiveness of lexicon-based extraction. We can see that adding the lexicon-based extraction scheme can improve the recall R, indicating that it indeed obtains more correct emotion-cause pairs. Although the precision P slightly decreases, the F-score F_1 still performs better than only extracting the top-1 pair in a document. Thus, lexicon-based extraction is an effective way to improve the pair extraction performance of our one-step approach.

[4] https://academiasinicanlplab.github.io

[5] Appendix A.2 reports the results with BERT encoder.

# 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.

Comparison on Extracting Multiple Pairs

We further compare the results on extracting multiple pairs in one document. We divide each fold's test set into two subsets: one subset contains documents having only one emotion-cause pair, and the other subset contains documents having two or more emotion-cause pairs.

Table 3 reports the comparative results on two subsets respectively. It can be seen that our model consistently outperforms INTER-EC on both subsets. Our one-step approach is relatively more effective for documents with more than one emotion-cause pair (over 13% F_1 improvement).

Results on Emotion Cause Extraction

We also provide the comparative results with recently-proposed methods for emotion cause extraction task: a rule-based method RB (Lee et al., 2010a), a traditional machine learning based method MULTI-KERNEL (Gui et al., 2016), and three neural methods CONVMS-MEMNET (Gui et al., 2017), CANN (Li et al., 2018), and RTHN (Xia et al., 2019). Note that all of them utilize *known* emotion clauses as model input. The top half of Table 4 reports their performance.

The bottom half of Table 4 shows the comparative results of methods without using known emotion clauses as model input. It clearly demonstrates

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.

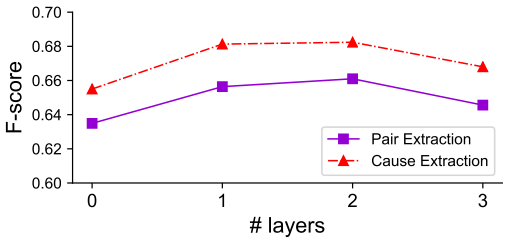

Figure 3: Results of RANKCP with various graph attention layers.

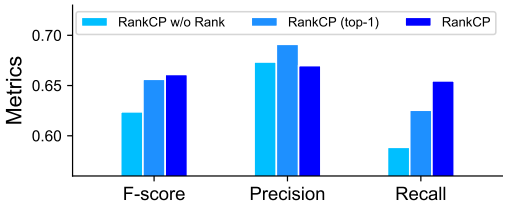

Figure 4: Comparative results of our variant model that removes the clause pair representation learning and ranking component (denoted as "RANKCP w/o Rank") and our full model RANKCP.

that our proposed RANKCP performs much better than other methods. Besides, although RANKCP does not utilize known emotions of test documents as model input, it still outperforms RB and MULTI-KERNEL, and is comparable to CONVMS-MEMNET. Thus, our approach benefits from inter-clause modeling and shows its effectiveness on cause clause extraction.

4.3 Further Discussions

We conduct ablation studies to analyze the effects of different components in our approach.

Effect of Two-level Supervision

Our model is trained with a mixture of two supervised signals: a low-level signal $\mathcal{L}_{\text{emo}} + \mathcal{L}_{\text{cau}}$ on clause representation learning at the output of graph attention network (see Eq. 5), and a high-level signal $\mathcal{L}_{\text{pair}}$ on clause pair representation learning and ranking (see Eq. 10). To verify the effect of low-level supervision, we train our model with $\mathcal{L}_{\text{pair}}$ only, and the results compared with those of our full model are given in Table 5. It shows that training with two-level supervision boosts the extraction performance. This indicates that incorporating a low-level supervision helps learn better clause representations, and eventually facilitates the clause pair representation learning and ranking process.

Effect of Graph Attention Layers

Graph attention network for modeling inter-clause latent relationships is the key component of our approach. We vary the number of graph attention layers (ranging from 0 to 3) to test its effect, and the results on emotion-cause pair extraction and cause clause extraction are shown in Fig. 3.

Obviously, the model without graph attention layer can not obtain good performance. Our approach achieves the best performance with two-layer graph attention network, indicating that inter-clause relationships can be modeled sufficiently without stacking a lot of layers in this task.

Effect of Clause Pair Representation Learning

We further investigate if we can obtain ideal performance by directly using clause representations to predict emotion clauses and cause clauses. In other words, we remove the clause pair representation learning and ranking component, and utilize the graph attention network's predictions (i.e., Eq. 5) to produce emotion-cause pairs. After predicting emotion clauses and cause clauses in a document, we consider all combinations of the predicted emotions and causes as the extracted emotion-cause pairs, and the comparative results of this variant model and our full model are shown in Fig. 4.

RANKCP performs much better than the variant one (especially on Recall), demonstrating that

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".

only offering clause-level predictions is not suitable for emotion-cause pair extraction task. Thus, combining clause-level and clause pair representation learning in a unified one-step model is indeed effective for extracting emotion-cause pairs.

Effect of Relative Position Embedding

We remove the relative position embedding part in RANKCP to verify its effect. We also compare vanilla and kernel-based relative position embedding schemes. The results are given in Table 6.

Removing relative position embedding results in performance degradation, indicating that relative position between a clause pair is indeed useful for prediction. Another observation from the first two lines is that lexicon-based extraction can not outperform top-1 extraction, which further verifies that the model without relative position embedding can not offer ideal ranking list. Kernel-based embedding achieves better performance than vanilla one on both top-1 and lexicon-based extractions, thus considering the mutual impact among relative positions helps obtain more powerful clause pair representations and further improves the performance of emotion-cause pair extraction.

4.4 Case Analysis

We illustrate a document that our approach RANKCP correctly extracts its emotion-cause pair (c_5, c_4) while INTER-EC fails:

4月11日 (c_1)，长沙网友洛丽塔在网上发帖吐槽 (c_2)，她有一个极品男友 (c_3)，如果要去的餐馆没有团购就要求换地方 (c_4)，这让她感觉很不爽 (c_5)，也很没面子 (c_6)。

Translation: *On April 11th* (c_1), *a netizen posted her complains on the Internet* (c_2), *she has a wacko boyfriend* (c_3), *he never goes to a restaurant without discounts* (c_4), *this makes her feel bad* (c_5), *and very embarrassed* (c_6).

We visualize the attention weights for two clauses c_4 and c_5 in Fig. 5. The emotion clause c_5 attends the corresponding cause c_4 with the highest weight, indicating that graph attention effectively captures the relationship between the two clauses.

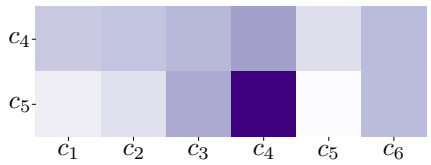

Figure 5: Attention weights for two clauses c_4 and c_5.

5 Related Work

Emotion Cause Extraction Lee et al. (2010a,b) first studied emotion cause extraction and designed a linguistic rule-based system to detect cause events. Early work attempted rule-based (Chen et al., 2010; Neviarouskaya and Aono, 2013; Gao et al., 2015), commonsense-based (Russo et al., 2011), and traditional machine learning based (Ghazi et al., 2015) approaches to extract causes for certain emotion expressions.

Gui et al. (2016) proposed an event-driven multi-kernel SVM method and released a benchmark corpus. Both feature based (Xu et al., 2019) and neural approaches (Gui et al., 2017; Li et al., 2018; Ding et al., 2019; Yu et al., 2019) have been proposed recently. Xia et al. (2019) adopted Transformer encoder augmented with position information and integrated global prediction embedding to improve performance. Fan et al. (2019) incorporated sentiment and position regularizers to restrain parameter learning. Hu et al. (2019) exploited external sentiment classification corpus to pretrain the model.

In other research lines, some work (Cheng et al., 2017) extracted emotion causes in the context of microblog with multi-user structure. Besides, Kim and Klinger (2018) and Bostan et al. (2020) addressed emotions as structured phenomena, and studied the semantic roles of emotions including trigger phrases, experiencers, targets and causes, as well as the reader's perception.

Emotion-Cause Pair Extraction All previous studies on emotion cause analysis need to take known emotion clauses as model input. The pioneer work (Xia and Ding, 2019) first put forward emotion-cause pair extraction task. They proposed a two-step approach to extract emotion and cause clauses separately, and then train a classifier to filter out negative pairs. Unlike their work, our work is a one-step solution for end-to-end emotion-cause pair extraction via effective inter-clause modeling, achieving significantly better performance.

6 Conclusion and Future Work

In this paper, we propose the first one-step neural approach RANKCP to tackle the problem of emotion-cause pair extraction, which emphasizes inter-clause modeling from a ranking perspective. Our approach effectively models inter-clause relationships to learn clause representations, and integrates relative position enhanced clause pair ranking into a unified neural network to extract emotion-cause pairs in an end-to-end fashion. Experimental results on the benchmark dataset demonstrate that RANKCP significantly outperforms previous systems, and further analysis verifies the effectiveness of each component in our model.

In future work, we shall explore the following directions. First, current studies on emotion cause analysis mainly focus on clause-level extraction which is relatively coarse-grained, and it is desirable to further design fine-grained methods that can extract span-level or phrase-level emotion expressions and causes. Second, designing effective methods to inject appropriate linguistic knowledge into neural models is valuable to emotion analysis tasks (Ke et al., 2019; Zhong et al., 2019). Finally, it would be interesting to study the semantic roles of emotion (Bostan et al., 2020), which considers the full structure of an emotion expression and is more challenging.

Acknowledgments

This work was supported in part by the Ministry of Science and Technology of China under Grants #2016QY02D0305 and #2018ZX10201001, and NSFC under Grants #71621002, #61671450 and #11832001. We thank the anonymous reviewers for their valuable comments.

References

Laura Bostan, Evgeny Kim, and Roman Klinger. 2020. GoodNewsEveryone: A corpus of news headlines annotated with emotions, semantic roles, and reader perception. In *LREC*.

Ying Chen, Sophia Yat Mei Lee, Shoushan Li, and Chu-Ren Huang. 2010. Emotion cause detection with linguistic constructions. In *COLING*.

Xiyao Cheng, Ying Chen, Bixiao Cheng, Shoushan Li, and Guodong Zhou. 2017. An emotion cause corpus for Chinese microblogs with multiple-user structures. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 17(1).

Robert De Beaugrande and Wolfgang U Dressler. 1981. *Introduction to text linguistics*. Routledge.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Zixiang Ding, Huihui He, Mengran Zhang, and Rui Xia. 2019. From independent prediction to reordered prediction: Integrating relative position and global label information to emotion cause identification. In *AAAI*.

Chuang Fan, Hongyu Yan, Jiachen Du, Lin Gui, Lidong Bing, Min Yang, Ruifeng Xu, and Ruibin Mao. 2019. A knowledge regularized hierarchical approach for emotion cause analysis. In *EMNLP-IJCNLP*.

Kai Gao, Hua Xu, and Jiushuo Wang. 2015. Emotion cause detection for Chinese micro-blogs based on ECOCC model. In *PAKDD*.

Diman Ghazi, Diana Inkpen, and Stan Szpakowicz. 2015. Detecting emotion stimuli in emotion-bearing sentences. In *CICLing*.

Lin Gui, Jiannan Hu, Yulan He, Ruifeng Xu, Qin Lu, and Jiachen Du. 2017. A question answering approach for emotion cause extraction. In *EMNLP*.

Lin Gui, Dongyin Wu, Ruifeng Xu, Qin Lu, and Yu Zhou. 2016. Event-driven emotion cause extraction with corpus construction. In *EMNLP*.

Jiaxing Hu, Shumin Shi, and Heyan Huang. 2019. Combining external sentiment knowledge for emotion cause detection. In *NLPCC*.

Pei Ke, Haozhe Ji, Siyang Liu, Xiaoyan Zhu, and Minlie Huang. 2019. SentiLR: Linguistic knowledge enhanced language representation for sentiment analysis. *arXiv preprint arXiv:1911.02493*.

Evgeny Kim and Roman Klinger. 2018. Who feels what and why? Annotation of a literature corpus with semantic roles of emotions. In *COLING*.

Terry Koo, Amir Globerson, Xavier Carreras, and Michael Collins. 2007. Structured prediction models via the matrix-tree theorem. In *EMNLP-CoNLL*.

Sophia Yat Mei Lee, Ying Chen, and Chu-Ren Huang. 2010a. A text-driven rule-based system for emotion cause detection. In *the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*.

Sophia Yat Mei Lee, Ying Chen, Shoushan Li, and Chu-Ren Huang. 2010b. Emotion cause events: Corpus construction and analysis. In *LREC*.

Xiangju Li, Kaisong Song, Shi Feng, Daling Wang, and Yifei Zhang. 2018. A co-attention neural network model for emotion cause analysis with emotional context awareness. In *EMNLP*.

Yang Liu and Mirella Lapata. 2018. Learning structured text representations. *Transactions of the Association for Computational Linguistics (TACL)*, 6.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *EMNLP-IJCNLP*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *ICLR*.

William C Mann and Sandra A Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text-interdisciplinary Journal for the Study of Discourse*, 8(3).

Daniel Marcu. 2000. *The theory and practice of discourse parsing and summarization*. MIT press.

Alena Neviarouskaya and Masaki Aono. 2013. Extracting causes of emotions from text. In *IJCNLP*.

Jiezhong Qiu, Jian Tang, Hao Ma, Yuxiao Dong, Kuansan Wang, and Jie Tang. 2018. DeepInf: Social influence prediction with deep learning. In *KDD*.

Irene Russo, Tommaso Caselli, Francesco Rubino, Ester Boldrini, and Patricio Martínez-Barco. 2011. EMOCause: An easy-adaptable approach to extract emotion cause contexts. In *WASSA@ACL-HLT*.

Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Training very deep networks. In *NIPS*.

W.T. Tutte. 1984. *Graph theory*. Encyclopedia of mathematics and its applications. Addison-Wesley Pub. Co., Advanced Book Program.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *ICLR*.

Shih-Ming Wang and Lun-Wei Ku. 2016. ANTUSD: A large Chinese sentiment dictionary. In *LREC*.

Rui Xia and Zixiang Ding. 2019. Emotion-cause pair extraction: A new task to emotion analysis in texts. In *ACL*.

Rui Xia, Mengran Zhang, and Zixiang Ding. 2019. RTHN: A RNN-Transformer hierarchical network for emotion cause extraction. In *IJCAI*.

Bo Xu, Hongfei Lin, Yuan Lin, Yufeng Diao, Liang Yang, and Kan Xu. 2019. Extracting emotion causes using learning to rank methods from an information retrieval perspective. *IEEE Access*, 7.

Xinyi Yu, Wenge Rong, Zhuo Zhang, Yuanxin Ouyang, and Zhang Xiong. 2019. Multiple level hierarchical network-based clause selection for emotion cause extraction. *IEEE Access*, 7.

Peixiang Zhong, Di Wang, and Chunyan Miao. 2019. Knowledge-enriched Transformer for emotion detection in textual conversations. In *EMNLP-IJCNLP*.

A Experimental Results with BERT

We employ pretrained BERT (Devlin et al., 2019) to replace the original encoder (Hierarchical RNN) in RANKCP, and report the experimental results.

A.1 RANKCP with BERT Encoder

Given a document $D = (c_1, c_2, \ldots, c_{|D|})$ where the i-th clause $c_i = (w_1^i, w_2^i, \ldots, w_{|c_i|}^i)$, to feed D into pretrained BERT, for each clause we insert a `[CLS]` token before it and append a `[SEP]` token to it, obtaining $c_i = (\texttt{[CLS]}, w_1^i, w_2^i, \ldots, w_{|c_i|}^i, \texttt{[SEP]})$. Following (Liu and Lapata, 2019), we use "interval" segment embeddings $(E_A, E_B, E_A, \ldots)$ to distinguish clauses in a document, i.e., E_A for clauses at odd positions and E_B for those at even positions. For each token in the document, its input representation is the sum of the corresponding token, segment, and position embeddings. The clause representation of clause c_i is the corresponding `[CLS]` token's output representation.

We implement our model based on `PyTorch` and `Transformers`,[6] and the BERT encoder is initialized using `BERT-Base, Chinese`.[7] The model is optimized by Eq. 13 for 20 epochs with early stopping, using AdamW optimizer (Loshchilov and Hutter, 2019) and 1e-5 learning rate. We schedule the learning rate that the first 10% of all training steps is a linear warm-up phrase and then a linear decay phrase is used.

A.2 Results

Table 7 shows the results on emotion-cause pair extraction and two sub-tasks. With the pretrained BERT encoder, the results of RANKCP significantly perform better than those with hierarchical RNN, especially on Recall, which indicates the effectiveness of contextualized embeddings as external knowledge, and thus pretrained BERT is a suitable backbone network for emotion-cause pair extraction task.

Table 8 shows the comparative results on extracting one and more than one pair, and we can observe that pretrained BERT encoder further improves the performance of RANKCP for extracting multiple pairs in one document.

[6] https://github.com/huggingface/transformers

[7] https://github.com/google-research/bert

Approach	Emotion-Cause Pair Extraction			Emotion Clause Extraction			Cause Clause Extraction		
	F_1	P	R	F_1	P	R	F_1	P	R
RANKCP	0.6610	0.6698	0.6546	0.8548	0.8703	0.8406	0.6824	0.6927	0.6743
RANKCP w/ BERT	0.7360	0.7119	0.7630	0.9057	0.9123	0.8999	0.7615	0.7461	0.7788

Table 7: Experimental results with pretrained BERT encoder.

# Pairs	Approach	F_1	P	R
One per doc.	RANKCP	0.6790	0.6625	0.6966
	w/ BERT	0.7633	0.7203	0.8123
Two or more per doc.	RANKCP	0.5531	0.7508	0.4390
	w/ BERT	0.5802	0.6772	0.5146

Table 8: Comparative results for documents with only one and more than one emotion-cause pair.

B More Discussions on Modeling Inter-Clause Relationships

B.1 Multi-Root Discourse Tree Induction

In previous experiments, we let RANKCP induce a discourse dependency tree (each discourse unit is a clause) while extracting emotion-cause pairs in a document. We expect that a document can be structurally represented as a multi-root dependency tree, where each root node is an emotion clause, and its child nodes plus the root itself are potential causes. To this end, we extended the original graph attention to a *structured graph attention* mechanism, inspired by (Koo et al., 2007; Liu and Lapata, 2018). See the next sub-section for details.

However, the structured graph attention does not lead to improvement for RANKCP. The main reason might be that dependencies in a discourse tree cannot handle a common situation well, i.e., an emotion clause and its corresponding cause clause is the same one. We leave the exploration of effective tree induction methods with the help of clause pair representation learning for future work.

B.2 Structured Graph Attention

At the t-th layer, let $\{\boldsymbol{h}_1^{(t-1)}, \boldsymbol{h}_2^{(t-1)}, \ldots, \boldsymbol{h}_{|D|}^{(t-1)}\}$ denote the input clause representations. The structured graph attention mechanism operates on each clause c_i via the following aggregation scheme:

$$\begin{aligned}
\boldsymbol{p}_i^{(t)} &= \alpha_i^{(t)} \boldsymbol{e}_{\text{root}} + \sum_{j \in \mathcal{N}(i)} \alpha_{ji}^{(t)} \boldsymbol{h}_j^{(t-1)}, \\
\boldsymbol{c}_i^{(t)} &= \sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(t)} \boldsymbol{h}_j^{(t-1)}, \\
\boldsymbol{h}_i^{(t)} &= \mathsf{ReLU}\left(\boldsymbol{W}_g^{(t)} \left[\boldsymbol{p}_i^{(t)}; \boldsymbol{c}_i^{(t)}; \boldsymbol{h}_i^{(t-1)}\right] + \boldsymbol{b}_g^{(t)}\right),
\end{aligned} \tag{15}$$

where $\boldsymbol{p}_i^{(t)}$ and $\boldsymbol{c}_i^{(t)}$ are the context information aggregated from parent clauses and child clauses respectively. $\alpha_{ij}^{(t)}$ reflects the marginal probability of a dependency between two clauses c_i and c_j. $\alpha_i^{(t)}$ denotes the probability of c_i being a root, and $\boldsymbol{e}_{\text{root}}$ is a special root embedding. Specifically, two MLPs compute unnormalized values $e_{ij}^{(t)}$ and $e_i^{(t)}$:

$$\begin{aligned}
e_{ij}^{(t)} &= {\boldsymbol{w}_d^{(t)}}^\top \tanh\left(\left[\boldsymbol{W}^{(t)}\boldsymbol{h}_i^{(t-1)}; \boldsymbol{W}^{(t)}\boldsymbol{h}_j^{(t-1)}\right]\right), \\
e_i^{(t)} &= {\boldsymbol{w}_r^{(t)}}^\top \tanh\left(\boldsymbol{W}^{(t)}\boldsymbol{h}_i^{(t-1)}\right),
\end{aligned} \tag{16}$$

Then, the normalized weights $\alpha_{ij}^{(t)}$ and $\alpha_i^{(t)}$ can be regarded as constrained attention weights to induce a non-projective discourse dependency tree based on Kirchhoff's Matrix-Tree Theorem (Tutte, 1984; Koo et al., 2007), where $\boldsymbol{A}^{(t)}$ and $\boldsymbol{L}^{(t)}$ denote adjacency matrix and Laplacian matrix respectively:

$$\begin{aligned}
[\boldsymbol{A}^{(t)}]_{ij} &= \begin{cases} 0, & \text{if } i = j, \\ \exp(\mathsf{LeakyReLU}(e_{ij}^{(t)})), & \text{otherwise}. \end{cases} \\
r_i^{(t)} &= \exp(e_i^{(t)}),
\end{aligned} \tag{17}$$

$$[\boldsymbol{L}^{(t)}]_{ij} = \begin{cases} \sum_{k=1}^{|D|} [\boldsymbol{A}^{(t)}]_{kj} & \text{if } i = j, \\ -[\boldsymbol{A}^{(t)}]_{ij}, & \text{otherwise}. \end{cases} \tag{18}$$

$$\hat{\boldsymbol{L}}^{(t)} = \boldsymbol{L}^{(t)} + \text{diag}(r_1^{(t)}, \ldots, r_{|D|}^{(t)}) \tag{19}$$

The normalized weights are:

$$\begin{aligned}
\alpha_{ij}^{(t)} &= (1 - \delta_{1,j})[\boldsymbol{A}^{(t)}]_{ij}[\hat{\boldsymbol{L}}^{(t)^{-1}}]_{jj} \\
&\quad - (1 - \delta_{i,1})[\boldsymbol{A}^{(t)}]_{ij}[\hat{\boldsymbol{L}}^{(t)^{-1}}]_{ji}, \\
\alpha_i^{(t)} &= r_i^{(t)}[\hat{\boldsymbol{L}}^{(t)^{-1}}]_{i1},
\end{aligned} \tag{20}$$

where δ is Kronecker delta and $\cdot^{-1}$ denotes matrix inversion. Eq. 19 is suitable for multi-root setting. In the case of single-root setting, it is replaced by:

$$[\hat{\boldsymbol{L}}^{(t)}]_{ij} = \begin{cases} r_j^{(t)} & \text{if } i = 1, \\ [\boldsymbol{L}^{(t)}]_{ij}, & \text{otherwise}. \end{cases} \tag{21}$$

During training, a cross-entropy loss is used to each layer's root probability $\alpha_i^{(t)}$, similar to $\hat{y}_i^{\text{emo}}$.