Optimizing Attention for Sequence Modeling via Reinforcement Learning

Hao Fei, Yue Zhang, *Member, IEEE*, Yafeng Ren, and Donghong Ji

***Abstract*— Attention has been shown highly effective for modeling sequences, capturing the more informative parts in learning a deep representation. However, recent studies show that the attention values do not always coincide with intuition in tasks, such as machine translation and sentiment classification. In this study, we consider using deep reinforcement learning to automatically optimize attention distribution during the minimization of end task training losses. With more sufficient environment states, iterative actions are taken to adjust attention weights so that more informative words receive more attention automatically. Results on different tasks and different attention networks demonstrate that our model is of great effectiveness in improving the end task performances, yielding more reasonable attention distribution. The more in-depth analysis further reveals that our retrofitting method can help to bring explainability for baseline attention.**

***Index Terms*— Attention mechanism, deep reinforcement learning (RL), natural language processing (NLP), neural networks.**

I. Introduction

THE attention mechanism has been applied to a range of natural language processing (NLP) tasks, such as neural machine translation [1], [2], dialog generation [3], machine reading/comprehension [4], sentiment classification [5], and text summarization [6]. It calculates the context representation of a sentence as a weighted sum of individual components, automatically selecting more important parts of an input sequence. This coincides with the psycholinguistic intuition to some extent, as humans often pay more attention to important parts to form a whole picture of sentences.

Many types of attention have been extensively used for NLP tasks, achieving competitive performances. Broadly speaking, attention can be divided into soft attention [2], [7] and hard attention [8], [9], according to whether or not entire elements of input sequence is used to compute the alignment scores [1]. In addition, self-attention (SA) gives strong results on a wide range of NLP tasks, in spite of its simplicity [10]–[12]. Hierarchical attention shows its power when dealing with documental level textual modeling [13].

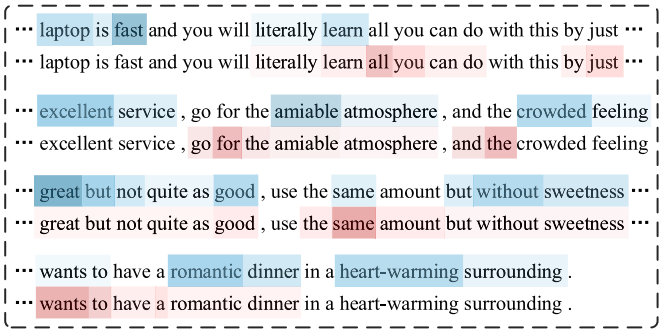

Fig. 1. Attention comparisons. Blue: ideal attention weights. Red: empirical attention weights. The color depth indicates the weight. Example sentences are partially displayed due to the space limitation.

Though bringing considerable improvements on various tasks in NLP, attention weights can fail to capture the most informative part of the content and do not always satisfy the needs of end tasks [14], [15]. Fig. 1 shows an example of fine-grained sentiment analysis.[1] Intuitively, the attention mechanism should properly highlight the elements that are task-related, as the examples marked in blue. Nevertheless, the vanilla attention mechanism does not generate proper attention for all the elements. In fact, a large proportion of the words with higher attention weights is not important or task-related, such as stop words, which are the examples visualized with red. Besides, according to a recent study [16], around 30% highlighted words by neural attention is not coincident with human attention. This negatively affects the final prediction of the model. Such a phenomenon is more serious for long sequences. Intuitively, the performance can be improved if the attention distribution is effectively fine-tuned and optimized.

Some methods use direct supervision to modify attention values [17], [18], but it can be highly costly and not feasible to obtain ideal attention signals for all tasks. In attention models, the weights obtained can be regarded as unsupervised signals learned during the optimization of the end task. To this end, reinforcement learning (RL) provides a practical tool that

Manuscript received 31 January 2020; revised 3 July 2020 and 1 November 2020; accepted 16 January 2021. Date of publication 10 February 2021; date of current version 4 August 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 61702121 and Grant 61772378, in part by the National Philosophy Social Science Major Bidding Project under Grant 11&zd189, in part by the Research Foundation of Ministry of Education of China under Grant 18JZD015, in part by the Key Project of State Language Commission of China under Grant ZDI135-112, and in part by the Guangdong Basic and Applied Basic Research Foundation of China under Grant 2020A151501705. *(Corresponding author: Yafeng Ren.)*

Hao Fei and Donghong Ji are with the School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, China.

Yue Zhang is with the School of Engineering, Westlake University, Hangzhou 310024, China.

Yafeng Ren is with the School of Interpreting and Translation, Guangdong University of Foreign Studies, Guangzhou 510420, China (e-mail: renyafeng@whu.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2021.3053633.

Digital Object Identifier 10.1109/TNNLS.2021.3053633

[1]The examples are sampled from SemEval2014/2015 data sets.

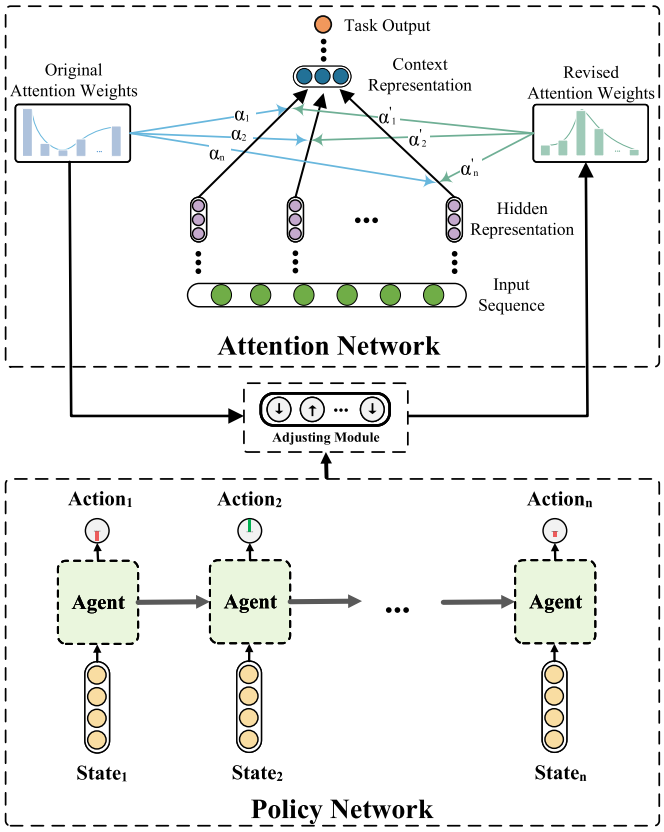

Fig. 2. Overall architecture of DRGA.

has been shown effective for various NLP tasks [19], [20]. In this article, we investigate the effectiveness of deep RL for obtaining better attention weights, by designing a novel sequential decision process that iteratively modifies original attention weights given a baseline attention network.

In particular, as shown in Fig. 2, given a baseline network, actions are dynamically taken to either increase or decrease the attention score of each node. A delayed reward is used to guide the learning of the policy network. With new context representations calculated from revised attention weights, the attention network can be optimized by backpropagation via a policy gradient-based learning algorithm. Supervision from end task loss at the global scope helps the model to revise the baseline attention weights. The baseline attention network fine-tuned by our RL framework during training can alone make a better prediction at the inference phase. Similar to the naive attention mechanism, the optimization of attention via our RL framework are derived solely from loss minimization of the end task, without using external guidance.

We compare RL-adjusted attention weights with the original attention weights on four types of attention mechanisms, including general attention [21], SA [10], hierarchical attention [13], and attention-based encoder–decoder [2]. Results on various benchmark data sets show that our proposed method brings strong performance gains in all different settings, yielding more intuitively reasonable attention distribution. Further in-depth analysis proves that the explainability of the attention mechanism largely depends on the exact attention architectures and the end tasks. Our retrofitting method can help to generate interpretable attention distribution and bring explainability for baseline attention. Our implementation codes are publicly available for research purpose on https://github.com/Baxelyne/DRGA under Apache License 2.0.

II. Related Work

A. Attention Mechanism

Attention showed its early potentials in NLP on neural machine translation (NMT) [1]. Attention mechanisms have subsequently been applied to various NLP tasks, including dialog generation [3], machine reading/comprehension [4], [22], sentiment analysis [23], [24], text summarization [6], machine translation [11], and information extraction [25], achieving state-of-the-art accuracies. For example, Wang *et al.* [26] propose an attention network incorporating both word- and clause-level attentions for aspect-based sentiment classification. Shen *et al.* [10] propose an SA-based contextual information CNN network for text classification. The transformer-style SA network employs multiple dimensions and multiple heads, resulting in boosted task performances [11], [12].

Recently, research efforts are paid to retrofit the vanilla attention mechanism. For example, Niculae and Blondel [17] use a smoothed max-operator as the replacement of softmax attention to give more interpretability without sacrificing the performance of the attention mechanism. Shen *et al.* [27] integrate both soft and hard attention into one context fusion model, "reinforced SA," for natural language inference and semantic relatedness tasks. In contrast, we let the attention module sufficiently and directly access the end task loss under both the global and the local scope, and calculate revised attention weights dynamically using an RL framework, to better capture informative elements. On the other hand, the capacity of attention representation of transformer has been much studied (e.g., the syntax-aware feature induction [28]–[30]). In this article, we show that the transformer attention can also be enhanced via our framework.

This work is also relevant to the recent hot topic that whether the attention mechanism carries interpretability power with intuition [16]. Some studies prove that the attention network can help to yield better task performances while inducing attention distribution with intuition [31]–[34]. Others argue that attention does not necessarily correspond to importance, and the explainability of the attention mechanism is denied [35], [36]. In this article, we try to give an answer to the question of attention explainability through our retrofitting RL framework.

B. Deep RL

This work also belongs to the application of deep RL for NLP. RL for NLP task recently attracts enormous interest [19], [20], [27]. For example, He *et al.* (2016) [37] use RL to fine-tune a bilingual machine translation model. Yu *et al.* (2017) [38] propose an RL-based reading method, which skims the insignificant slots to achieve higher time efficiency. Yin *et al.* (2018) [19] introduce a deep RL framework for the Chinese zero pronoun resolution.

Some work employs RL to enhance machine learning methods rather than certain tasks. For example, Fang *et al.* [39] investigate RL for active learning, Wu *et al.* [40] investigate RL for cotraining, and Chen *et al.* [41] investigate RL for self-training. Our work more closely relates to the work of Indurthi *et al.* [42], who exploit deep RL over baseline hard-attention mechanisms for improving NMT on long sequences [42].

In this article, we employ policy gradient for sampling actions, which can maintain the exploration and avoid getting stuck at an intermediate state [43], [44]. For continuous control of continuous attention weights, we use the probabilistic policy gradient algorithm [45] with action sampling from a Gaussian distribution.

III. Framework

We formulate attention distribution amendment as a sequential decision process and employ RL since: 1) it provides a sequential-decision scheme at the token level for dynamical optimization without any supervision and 2) it takes into account sufficient environment states when making every decision. Our RL method uses the environment as a consultant at each time step to dynamically revise weights, by modeling weight-assigning as a sequential decision process, which can be more intuitive and reasonable.

The architecture of our method is shown in Fig. 2. We name the framework as deep RL guided attention (named DRGA). It consists of three components: an attention network, a policy network, and an adjusting module. The attention network computes initial attention scores for an input sequence. Then, the RL agent generates modification actions according to the environment states. Finally, the adjusting module revises the weights according to the action values. The revised attention weights will be used for the attention network to calculate attention context representation for making prediction of a specific task. The policy network obtains rewards from the attention network's prediction, which, in return, guides the learning of the policy for adjusting attention weights.

A. Policy Network

Formally, we design the RL agent as policy network $\pi_\theta(s, a) = P(a|s; \theta)$, where s stands for the state, a represents the action, and θ indicates the parameters of the model. Instead of using a deep Q-network, which learns a greedy policy for discrete actions, we employ the policy gradient algorithm [44] for sampling actions, due to the need for continuous control of attention weights. We adopt a stochastic policy, sampling actions with probabilities from a Gaussian distribution at each step, which can maintain exploration and prevent the agent from getting stuck at an intermediate state.

1) State: At each time step $t \in \{1, 2, \ldots, n\}$, a state $\boldsymbol{s}_t$ consists of the word vector $\boldsymbol{w}_t \in \mathbb{R}^{D_w}$ of the current element, the hidden representation $\boldsymbol{h}_t \in \mathbb{R}^{D_h}$ of the current element, a weighted representation $\boldsymbol{h}_t^* \in \mathbb{R}^{D_h}$ of the current element, the sum of the original weighted element $\boldsymbol{h}^s \in \mathbb{R}^{D_h}$, the concatenation of all element representation $\boldsymbol{h}_t^a = [\boldsymbol{h}_1; \ldots; \boldsymbol{h}_n] \in \mathbb{R}^{n*D_h}$, and the query representation $\boldsymbol{u} \in \mathbb{R}^{D_u}$. The word vector $\boldsymbol{w}_t$ can be obtained from a lookup table, and the hidden representation $\boldsymbol{h}_t$ can be obtained via a sequential network, e.g., recurrent neural network. The word vector $\boldsymbol{w}_t$ and the hidden representation $\boldsymbol{h}_t$ of the current element provide basic information of the element itself. The weighted representation of the current element $\boldsymbol{h}_t^* = \alpha_t * \boldsymbol{h}_t$ is the dot product of the original weight and the hidden representation, alternatively. The weighted sum $\boldsymbol{h}^s = \sum_t \alpha_t * \boldsymbol{h}_t$, together with the query representation $\boldsymbol{u}$, offers the global context information. Formally, the state for the policy network is defined as follows:

$$\boldsymbol{s}_t = \left[\boldsymbol{w}_t; \boldsymbol{h}_t; \boldsymbol{h}_t^*; \boldsymbol{h}^s; \boldsymbol{h}_t^a; \boldsymbol{u}\right]. \tag{1}$$

Thus, given a sentence (a sequence of tokens) $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$, there will be a corresponding sequence of environment states $\boldsymbol{S} = \{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_n\}$. $\boldsymbol{S}$ is fed into the agent for generating the corresponding action a_t.

2) Action and Adjusting Strategy: Given a sequence of states $\boldsymbol{S}$, a corresponding action sequence $\boldsymbol{A} = \{a_1, \ldots, a_n\}$ is sampled based on the probability density of the Gaussian distribution [43], [45], [46]. The output of the agent is a description of the Gaussian distribution for the action

$$\begin{aligned} \pi(s_t; \theta) &= \mathcal{N}(\mu(s_t, \theta), \quad \sigma(s_t, \theta)) \\ a_t &\sim \pi(s_t; \theta) \end{aligned} \tag{2}$$

where μ and σ are the mean and standard deviation. We employ a layered feedforward network (FFN) to output parametric policy, from which the unary action will be sampled. We then squash the action between $[-1, 1]$: $a_t \leftarrow \text{sigmoid}(a_t) - 1$.

The agent keeps sampling for the whole sentence. Once all the actions for the input sequence are decided, the adjusting module increases or decreases the corresponding attention score based on the action values. Specifically, a new weight score is the sum of the original value and the corresponding action value. Therefore, if an action value $a_i > 0$, the resulting score will be increased, and vice versa. To make the agent more flexible to the situation, we adopt a trainable parameter β ($0 < \beta \leq 1$) as the scale. β is initialized with a specific value and decreases gradually as training progress goes forward.

Specifically, given an original attention vector $\boldsymbol{\alpha} = \{\alpha_1, \ldots, \alpha_n\}$ and the actions $\boldsymbol{A} = \{a_1, \ldots, a_n\}$, the adjustment is defined as follows:

$$\alpha_i^* = \begin{cases} \alpha_i + a_i * \beta, & \alpha_i^* > 0 \\ e^{-\text{INF}}, & \alpha_i^* \leq 0. \end{cases} \tag{3}$$

Then, we linearly normalize the attention values to ensure equal-to-one sum

$$\boldsymbol{\alpha}_t' = \frac{\boldsymbol{\alpha}_t^*}{\sum \boldsymbol{\alpha}_{t'}^*}. \tag{4}$$

The revised attention scores $\boldsymbol{\alpha}' = \{\alpha_1', \ldots, \alpha_n'\}$ are passed to the attention network.

3) Reward: Once the attention network makes a prediction based on the revised attention representation, the posterior output probability $P(y|\boldsymbol{X})$ is computed. Note that the policy network cannot update its parameters after each action. Instead, updates are executed only after the entire sequence of elements is processed. Therefore, a delayed reward is used to guide the policy learning, where $P(y|\boldsymbol{X})$ is used by the

policy network to compute reward R_L for leading to task-relevant attention. To obtain a delayed reward based on the attention network's prediction, we perform action sampling over the entire sequence.

The reward is crucial for the RL agent to optimize its policy. To encourage the agent to make proper adjustments, we add an additional term to regulate the number of highly weighted elements. Considering that the salient words in a sequence should be neither too many nor too few, we employ a unimodal function $f(x) = x + L_0/x$, which has a minimum value at $x_0 = \sqrt{L_0}$. Thus, the reward is

$$R_L = \log P(y|\boldsymbol{X}) - \gamma\,(L/N + N \cdot L_0/L) \tag{5}$$

where L denotes the length of the sequence, N is the number of highly weighted elements,[2] and γ is a harmonic factor to balance the two parts. The second term encourages N to be $\sqrt{L_0} \cdot L$, which means that, for instance, when $L_0 = 0.01$, a sequence with a length of $L = 10$ has around one salient element.

4) Objective: We train the policy gradient agent using the REINFORCE algorithm [47], which aims to maximize the expected reward

$$\begin{aligned} J(\Theta) &= E_{(s_t,a_t)P_\theta(s_t,a_t)} R(\boldsymbol{s_1} a_1 \cdots \boldsymbol{s_T} a_T) \\ &= \sum_{s_1 a_1 \cdots s_T a_T} \prod_t p(a_t|\boldsymbol{s}_t;\theta) R_L \end{aligned} \tag{6}$$

where $p(a|\boldsymbol{s};\theta)$ indicates the probability of a generated action a. Thereafter, we apply the likelihood ratio for calculating the gradients of the policy network

$$\nabla_\Theta J(\Theta) = \sum_t R_L \nabla_\Theta \log p(a|\boldsymbol{s};\theta). \tag{7}$$

In our work, the policy gradient provides timely direct supervision from the end task for each time-sequential modification. Also, the policy gradient enables the additional reward to function and the continuous control of actions.

B. Attention Network

The attention network is used to: 1) produce initial attention weights (a probability distribution) for a sequence of elements and 2) make a prediction for a specific task given a final attention based representation.

The task-specified network takes an input sequence $\boldsymbol{X} = \{x_1, \ldots, x_n\}$ and obtains word embeddings $\boldsymbol{w_i} (1 < i < n)$. The network then encodes them into hidden representation $\boldsymbol{h}_i \in \{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_n\}$. A task-related vector representation of a query representation is given as $\boldsymbol{u} \in \mathbb{R}^{D_u}$. Thereafter, $\boldsymbol{h}_i$ is passed to the attention module to obtain a weighted representation $\boldsymbol{c}$ (also $\boldsymbol{h}^s$) and attention weights α, via the following equations:

$$v_i = f(\boldsymbol{u}, \boldsymbol{h}_i) \tag{8}$$

$$\boldsymbol{\alpha} = \text{softmax}(\boldsymbol{v}) \tag{9}$$

$$\boldsymbol{c} = \sum_i \alpha_i \boldsymbol{h}_i. \tag{10}$$

The weights $\boldsymbol{\alpha}$ are then adjusted by the agent of DRGA.

[2]The tokens are first sorted by their attention values within the sequence from the largest to the smallest, before the accumulated values are calculated token by token. An accumulated threshold δ is set based on the specific data set, and those tokens beneath the threshold are taken as highlighted ones.

In some cases, models involve multihead [11], multilayer [48], or multidimension [11] attention. For multilayer and multihead, attention weight is a $K \times N$ matrix (K is the number of layers or heads and N is the attention length), which can be regarded as K independent attention arrays. DRGA adjusts the K attention arrays one by one. For multidimensional attention, it is an $H \times N$ array. Each element $\vec{\boldsymbol{\alpha}}_{\boldsymbol{i}}$ in attention is an H-dimension vector. DRGA revises the vector via $\vec{\boldsymbol{\alpha}}_i^* = \vec{\boldsymbol{\alpha}}_i + \vec{\boldsymbol{a}}_i * \beta$, where $\vec{\boldsymbol{a}}_i \in \mathbb{R}^H$ is the corresponding action vector.

The attention network calculates the context representation $\boldsymbol{c}'$ using the revised attention weights $\boldsymbol{\alpha}'$ and hidden representation $\boldsymbol{h}_i$ via (10). The predicted probability $P(y|\boldsymbol{X})$ of a task can be output based on the context representation

$$P(y|\boldsymbol{X}) = \text{softmax}(f_{\text{net}}(\boldsymbol{c}')) \tag{11}$$

where $f_{\text{net}}(\cdot)$ denotes a specific network used.

C. Training

A cross-entropy loss function is employed to train the attention network

$$L = -\sum_{X \in D} \sum_1^K \hat{p}(y, \boldsymbol{X}) \log P(y|\boldsymbol{X}) \tag{12}$$

where $\hat{p}(y, \boldsymbol{X})$ is the gold distribution of $\boldsymbol{X}$.

In our framework, the attention network can be equipped with different types of attention implementation. We will elaborate on the details for different attention in the next section.

The policy network and the attention network are jointly trained. However, directly training the whole framework with cold-start would be difficult. Recently, pretraining has been proven crucial in RL. Thus, we pretrain the policy network and the attention network separately. In particular, we first pretrain the attention network until it is close to convergence. Then, we pretrain the policy network while keeping the parameters of the other model fixed. Finally, we jointly train all the components. The training process is demonstrated in Algorithm 1.

IV. Experiments

A. Experimental Setup

We conduct experiments on four typical attention architecture, including SA [10], task-specific attention [21], hierarchical attention [13], and attention-based encoder–decoder models [2]. Experiments are performed on three tasks, including fine-grained sentiment analysis, text classification, and NMT. The performances of classification tasks are compared and reported in accuracy.[3] We test the performances of our method ten times on all the corresponding test sets, and all the results are presented after Significance Test with $p \leq 0.015$. In our experiments, the quantity and the split setting (into train/dev/test subset) on each benchmarks data sets strictly follow the corresponding baselines.

[3]To make fare comparisons with baselines, we use only the accuracy measurement, which is the conventionally employed in baselines.

Algorithm 1 Training Procedure for DRGA

Require:

1) Pretrain the attention network via Eq. (12);
2) Fix the parameters of the attention network and pretrain the agent via Eq. (7);
3) Co-train the overall framework under following steps until convergence.

Ensure:

for each input sequence s_i **do**
  Feed the $\boldsymbol{X}$ into attention network;
  Compute attention weights α for s_i by attention network via Eq. (8) and Eq. (9);
  for each time step t in sequence **do**
    Prepare the states $\boldsymbol{s}_t$ in Eq. (1);
    Sample an action a_t via Eq. (2);
  end for
  Adjust original attention values by Adjusting module via Eq. (3) and Eq. (4);
  Calculate the new attention context representation for attention network via Eq. (10);
  Make prediction in attention network, and minimize the loss in Eq. (12);
  Compute reward for RL agent via Eq. (5);
  Update the parameters for both attention network and RL agent via Eq. (12) and Eq. (7).
end for

We employ the simple MLP as the RL agent. We set the dimension of both hidden states of the agent and word vectors as 300. The word vectors are initialized with Glove pretrained embeddings.[4] We use Adam [49] with a learning rate of 0.0005. Dropout is used for the attention network with a 0.5 keeping rate. All experiments are conducted using an Intel Core i7 7700HQ CPU with 8-GB memory. We exploit different coefficient factors for different tasks and data sets after fine-tuning the corresponding development sets.

B. Baseline Attention

1) SA: In (8), for SA, the query $\boldsymbol{u}$ stems from the input sequence itself, that is, $\boldsymbol{e}_i = f(\boldsymbol{h}_i, \boldsymbol{h}_i)$. SA has the capability of modeling the dependencies between tokens from the same sequence. We validate DRGA with five types of networks, including LSTM [50], CNN [51], CICNN [52], DiSAN [10], and transformer[5] [11], respectively. Besides, we also compare our model with a differentiated attention learning model by Zhou *et al.* [18].

The experiments for SA are conducted on four benchmark data sets for text classification, including Movie Review (MR) [53], Stanford Sentiment Treebank (SST) [54], Subjectivity (SUBJ) [55], and AGnews [56].

2) Task-Specific Attention: We test DRGA on general attention-based networks, in which the query $\boldsymbol{u}$ of the attention module is task-specific. We choose three attention-based networks for fine-grained sentiment analysis, including LSTM [57], IAN [21], and W&C-ATT [26]. The experiments are based on the data sets of SemEval2014 Task 4 (3-class) and SemEval2015 Task 12 (2-class), respectively. Both include two domains: laptop and restaurant.

3) Hierarchical Attention: For validating the effectiveness of DRGA on hierarchical attention, we choose two types of networks as baselines. The experiments are based on three benchmark data sets on document level text classification, including IMDB (10-class) and Yelp Data Set Challenge (5-class) in 2013 and 2014. We compare the performance of different models, including UPA [48] and HUAPA [13].

4) Attention-Based Encoder–Decoder: To evaluate the compatibility of DRGA on sequence generation, we perform experiments on the encoder–decoder structure. The experiments are conducted on the NMT task, based on the corpora of WMT14 English–German and WMT15 English–German, which are English to German translation data sets. We first use a typical attention-based seq2seq (Att-seq2seq) model proposed by Luong *et al.* [2] as our baseline. Besides, we employ transformer [11] and Hard-Attention (HardAtt) [42] for NMT tasks. The performance is reported in perplexities (Ppl) and BLEU.

TABLE I

RESULTS WITH DIFFERENT SETTINGS OF RL AGENT ON LSTM + SA WITH DRGA. # LAYER DENOTES THE TOTAL LAYERS OF MLP. # HIDDEN DENOTES THE SIZE OF HIDDEN LAYER. # PARAM DENOTES THE TOTAL NUMBER OF TRAINABLE PARAMETERS

# Layer	# Hidden	# Param	# Iter	Acc.
1	100	3,315K	920	93.8
	200	3,446K	1,228	94.3
	300	3,597K	1,420	94.9
2	100	3,336K	1,120	94.8
	200	3,527K	1,400	95.0
	300	3,778K	1,560	**95.5**
	400	4,089K	1,725	94.9
3	100	3,356K	1,210	94.0
	200	3,607K	1,500	94.5
	300	3,958K	1,630	94.9
LSTM+SA		3,212K	780	93.5

C. Development Experiments

We conduct several development experiments on the SUBJ development data set, based on the LSTM + SA with DRGA.

1) Regulating Factors: In our framework, the adjusting scale β is designed for the adjusting increment, and the harmonic factor γ is used for balancing the RL agent reward. They can be properly initialized for better performance. We compare the performances of DRGA under different combinations of β and γ by development experiments. Besides, β regulates the adjusting rates; therefore, we investigate the impact of β on the converging time.

From the results shown in Fig. 3, we can find that γ influences the performances of DRGA since it controls the number of highly weighted tokens. When γ is set as 0.2, DRGA achieves the best accuracy. An initial value of 0.4 gives the best performance. Besides, when the initial value of β is above 0.4, the time for the model to converge is shortened to around 50 min.

[4] http://nlp.stanford.edu/projects/glove/

[5] We only use the transformer encoder for classification tasks.

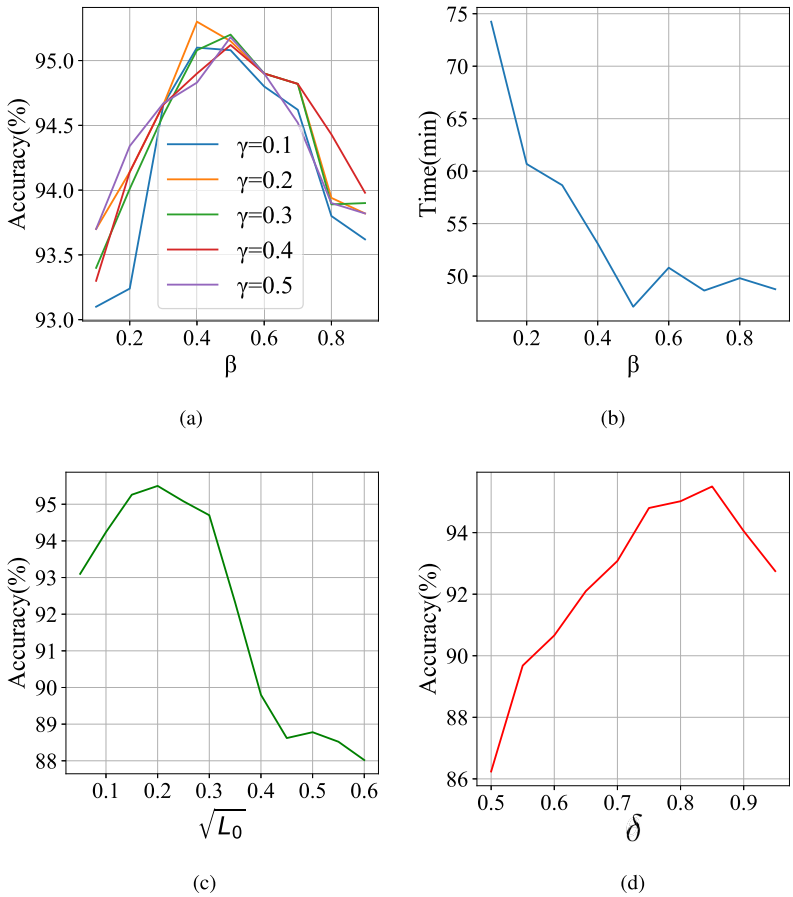

Fig. 3. Influences of regulating factors. (a) Accuracy over β and γ. (b) Training time over β. (c) Accuracy over $\sqrt{L_0}$. (d) Accuracy over δ.

TABLE II

RESULTS OF SA WITH DRGA. IN EACH GROUP, "+DRGA" MEANS INTEGRATING THE ATTENTION-BASED NEURAL NETWORK WITH DRGA. RESULTS OF BASELINE METHODS WITH MARK * ARE OBTAINED BY RUNNING THEIR RELEASED SOURCE CODE. OTHERS ARE REPRINTED FROM THE ORIGINAL PAPER

System	MR	SST	SUBJ	AGnews
LSTM*	77.4	46.4	92.2	90.9
LSTM+SA*	80.1	48.3	93.5	91.1
+DRGA	**82.4**	**50.6**	**95.5**	**92.8**
CNN*	81.5	48.0	93.4	91.6
CNN+SA*	81.9	48.5	93.3	91.5
+DRGA	**82.3**	**49.5**	**95.6**	**92.2**
CICNN	81.9	48.8*	93.8	88.4
+DRGA	**82.6**	**49.0**	**95.5**	**91.8**
DiSAN	**82.9***	51.7	94.2	92.3*
+DRGA	82.2	**52.7**	**95.5**	**93.3**
Transformer*	81.5	48.2	94.0	92.0
+DRGA	**82.4**	**49.9**	**95.8**	**92.9**
Zhou et al.	81.2*	48.8	94.1	91.5*

TABLE III

RESULTS OF TASK-SPECIFIC ATTENTION WITH DRGA

System	SemE-2014		SemE-2015		Avg.
	lap	rest	lap	rest	
LSTM*	66.5	74.3	73.5	73.4	71.9
ATAE	68.7	77.2	75.2	74.7	73.9
+DRGA	**70.0**	**78.1**	**77.0**	**76.2**	**75.3**
IAN	72.1	78.6	75.3	**75.5**	75.4
+DRGA	**74.7**	**80.8**	**77.1**	75.3	**77.0**
W&C-ATT	76.2	82.1	80.9	**81.6***	80.2*
+DRGA	**76.9**	**84.3**	**82.8**	81.5	**81.4**

TABLE IV

RESULTS OF HIERARCHICAL ATTENTION WITH DRGA

System	IMDB	Yelp 2013	Yelp 2014	Avg.
UPA	53.3	65.0	**66.7**	61.6
+DRGA	**55.1**	**66.2**	66.5	**62.6**
HUAPA	55.0	**68.3**	68.6	63.9
+DRGA	**56.2**	67.5	**68.8**	**64.2**

We introduce L_0 to guide the learning of salient words number and the accumulated threshold δ for filtering highly weighted elements. They should be decided according to the development data set. From the figure, we can see that, with $\sqrt{L_0} = 0.2$ and $\delta = 0.85$ on the SUBJ data set, DRGA achieves the best result.

The above analysis shows that these regulating factors should be fine-tuned for DRGA to reach the best performance on different data sets for different tasks.

2) Settings of RL Agent: To avoid complex optimization, we employ the light-weighted FFN as the RL agent. We now

TABLE V
RESULTS ON NMT TASKS

System	WMT14		WMT15	
	Ppl	BLEU	Ppl	BLEU
Att-seq2seq	6.3	23.0	9.7	25.9
+DRGA	**5.2**	**25.6**	**8.0**	**27.7**
Transformer	4.3	28.4	7.2*	29.8*
+DRGA	**3.7**	**29.8**	**6.0**	**31.9**
HardAtt	4.0	29.2	7.1*	30.2*

explore the performances of different numbers of parameters and the FFN topology of the RL agent. From the results shown in Table I, we can see that more parameters or more hidden layers do not always improve the performance. By using a two-layer FFN with 300-dim hidden size, the RL agent gives the best performance while keeping a comparably simple architecture. Overall, we see that a simple light-weighted MLP network can bring good improvements for baseline attention.

D. Quantitative Results

The final experimental results are shown in Tables II–V. For the task of text classification on data sets with SA, both LSTM and CNN achieve competitive results, as shown in Table II. By integrating DRGA, the performances of almost all the SA baselines are improved on each data set. These results demonstrate the effectiveness of the DRGA for optimizing the SA. We also compare our model with Zhou *et al.* (2018), which use two-branch architecture to shift the attention to different parts of a sentence. Our proposed method achieves better accuracies compared with the method of Zhou *et al.* (2018). Table III shows the results for fine-grained sentiment analysis. Similar observations are found using DRGA for task-specified attention networks. DRGA improves the accuracy of the task in three baselines for four data sets.

As shown in Table IV, DRGA improves the UPA model on the IMDB data sets by 1.8% and the Yelp 2013 data sets by 1.2% and helps HUAPA by 1.2% on IMDB and by 0.2% on Yelp 2014. We find that the improvement of DRGA for hierarchical attention is not as significant as that for SA and generic attention. The possible reason is the complexity of hierarchical attention, which has its own iterative modifications. Nevertheless, DRGA still brings overall improvement on the majority of data sets. Table V shows the results of attention-based seq2seq with DRGA. DRGA gives the same level of improvement for sequence generation models, helping the Att-seq2seq by 2.6% BLEU on WMT14, 1.8% BLEU on WMT15, and transformer by 1.2% and 2.1% BLEU.

E. Qualitative Results

We present qualitative analysis on how DRGA optimizes the attention mechanism by visualizing the attention weights α. The weights from the original attention are compared with the revised one by DRGA.

Table VI shows the results. First, we observe that the attention distribution revised by DRGA is more reasonable compared with the original attention model. Intuitively, when the original attention weight is generally functional, DRGA guides the attention module to give higher attention values to more task-related tokens, which consequently improves the performances. Attention weights of stop words, which are less informative for the task, are effectively weakened by DRGA. The optimization effect is more evident for long sentences due to the restraint of the additional term of reward, as described in subsection of **Reward**.

Incorrectly assigned higher attention values from the original attention module are properly revised by DRGA thanks to its ability to sufficiently interacting with the sentence environment. This can be found in the example of transformer[6] for classification task in Table VI. Informative phrases, such as best, artful large, and soon, are expected to have larger weights. While the transformer fails to achieve this, DRGA highlights the relevant words. Therefore, the attention distribution is much more concise and accurate. Besides, we track down the transition of attention weights, as shown in Table VII. We can find that the transitions from the original attention weights to final revised weights are stable and continuous, and some important words for the task receive increasingly more proper attention when the iteration increases. The above visualization shows the effectiveness of the proposed model.

F. Revisiting Informative Elements

DRAG can quantitatively improve the end task performances by reassigning the weights. To see how many sentences are optimized, we compared the original highlighted token numbers from the attention module and the highlighted token numbers that are revised by DRGA, in a fine-grained scope. We make comparisons between LSTM + SA and DRGA on four data sets. The results are shown in Table VIII. We find that, for each data set, DRGA decreases the count of originally highlighted words, with a maximum reduction rate of 23.8% on the SST data set and 18.3% on the AGnews data set.

To understand what types of information DRGA prefers to penalize or increase, we investigate the top-25 words whose weights are modified the most by DRGA. The results are shown in Fig. 4. We find that: 1) parts of the most highly selected words are stop words, such as a, that, and with, which can be uninformative for the task and 2) DRGA adjusts the weights' distribution for frequently highlighted tokens to a more reasonable proportion. Thus, salient words, such as interesting, great, and best, are further highlighted.

G. Attention Explainability

Finally, we try to investigate the explainability of the attention mechanism. The attention interpretability can be expressed as follows: if the learned attention weights agree with natural measures of features importance, the alternative of counterfactual attention distribution will correspondingly change the model output distribution most [34]–[36]. Under such assumption, following Serrano and Smith [36],

[6]The attention weights of the transformer are multidimensional, and we squash them into scalar values when we visualize them.

TABLE VI

VISUALIZATION OF ATTENTION WEIGHTS FOR BASELINE ATTENTION MODELS BEFORE AND AFTER THE OPTIMIZATION FROM OUR FRAMEWORK

System	Attention Distribution
self attention	
CICNN	contrived , awkward and filled with unintended laughs , the film shows signs that someone other than the director got into the editing room and tried to improve things by making the movie go faster .
+DRGA	contrived , awkward and filled with unintended laughs , the film shows signs that someone other than the director got into the editing room and tried to improve things by making the movie go faster .
Transformer	the movie is one of the best examples of artful large format filmmaking you are likely to see anytime soon .
+DRGA	the movie is one of the best examples of artful large format filmmaking you are likely to see anytime soon .
task-specific attention	
ATAE	The computer runs very fast with no problems and the iLife software that comes with it (iPhoto, iMovie, iWeb, iTunes, GarageBand) is all very helpful as well.
+DRGA	The computer runs very fast with no problems and the iLife software that comes with it (iPhoto, iMovie, iWeb, iTunes, GarageBand) is all very helpful as well.
hierarchical attention	
HUAPU	my friend suggested that i see this movie . i took his advise and am very glad that i did go and see it . not only does this movie have an entertaining plot , but it explores a lot of things dealing with human existence , such as free will , memories , the soul , and even hope and happiness .
+DRGA	my friend suggested that i see this movie . i took his advise and am very glad that i did go and see it . not only does this movie have an entertaining plot , but it explores a lot of things dealing with human existence , such as free will , memories , the soul , and even hope and happiness .

TABLE VII

TRANSITION OF ATTENTION WEIGHTS BY TANSFORMER + DRGA BASED ON THE EXAMPLE SENTENCE FOR CLASSIFICATION TASK

# Iter	the	movie	is	one	of	the	best	examples	of	artful	large	format	filmmaking	you	are	likely	to	see	anytime	soon	.
0	0.0079	0.0039	0.1572	0.2469	0.1169	0.0056	0.0028	0.0112	0.0076	0.0045	0.0021	0.0029	0.0509	0.1322	0.0529	0.1419	0.0028	0.0126	0.0117	0.0054	0.0076
500	0.0081	0.0096	0.0705	0.2092	0.0538	0.0081	0.0892	0.0492	0.0077	0.0792	0.0598	0.0525	0.0432	0.0541	0.0302	0.0702	0.0078	0.0093	0.0292	0.0492	0.0094
1,800	0.0292	0.0260	0.0351	0.1038	0.0284	0.0438	0.1238	0.0638	0.0260	0.1338	0.1164	0.0371	0.0258	0.0237	0.0248	0.0148	0.0318	0.0293	0.0188	0.0378	0.0303
2,500	0.0092	0.0077	0.0093	0.0078	0.0091	0.0862	0.1762	0.1062	0.0073	0.2062	0.0788	0.0695	0.0582	0.0106	0.0065	0.0083	0.0067	0.0088	0.0431	0.0802	0.0091

TABLE VIII

COMPARISONS OF HIGHLIGHTED TOKEN NUMBERS BETWEEN BEFORE AND AFTER DRGA

Dataset	Sent.	Before	After	Δ
MR	10,662	847	653	194
SST	11,855	1,305	963	342
SUBJ	10,000	1,054	803	251
AGnews	127,600	10,232	8,361	1,871

we explore the relative importance when the attention distribution has been changed. Using i to denote the token with the highest attention α_i in the sentence, we compare how i influence the model's output distribution, by comparing the importance of i with the other random attended element r, which is drawn uniformly from the same sentence. Concretely, we measure the difference of two Jensen–Shannon (JS) divergences

$$\Delta\text{JS} = \text{JS}(\boldsymbol{p}, \boldsymbol{q}_{\{i\}}) - \text{JS}(\boldsymbol{p}, \boldsymbol{q}_{\{r\}}) \tag{13}$$

where $\text{JS}(\boldsymbol{p}, q_{\{i\}})$ is the JS divergence of the model's original output distribution $\boldsymbol{p}$ and the output distribution $\boldsymbol{q}_{\{i\}}$ after removing the learned most important token i in attention, and the second JS divergence is the counterpart after removing the r in its attention. Intuitively, if i is truly the most important element, the change of output distribution by removing i can be the greatest, and correspondingly, we can expect ΔJS to be positive. Based on several different attention architectures and data sets, we plot ΔJS against $\Delta\alpha = \alpha_i - \alpha_r$.

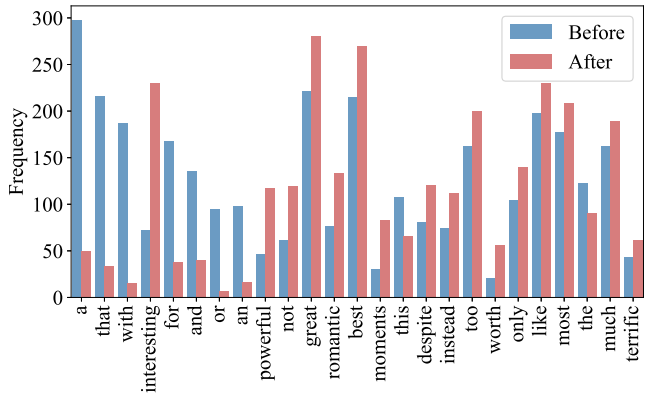

Fig. 4. 25 most modified words on SUBJ.

As shown in Fig. 5, several interesting patterns can be found. First, for all the vanilla attention, only $\Delta\alpha$ larger than 0.6 can activate above-zero ΔJS, while $\Delta\alpha$ threshold is around 0.2 in the attention improved by DRAG, which means that the retrofitted attention network can help to correctly adjust the most intuitive token i. Second, the same attention in different data sets can yield distinct capability of explainability, which can be found in Fig. 5(a) and (b), where DiSAN has different $\Delta\alpha$ threshold on the SST and SUBJ data sets. We can also notice that vanilla attentions can lead to negative ΔJS [e.g., in Fig. 5(a)], indicating task-unrelated attention weights. Finally, our DRAG framework is more useful in optimizing sequence-to-sequence attention in

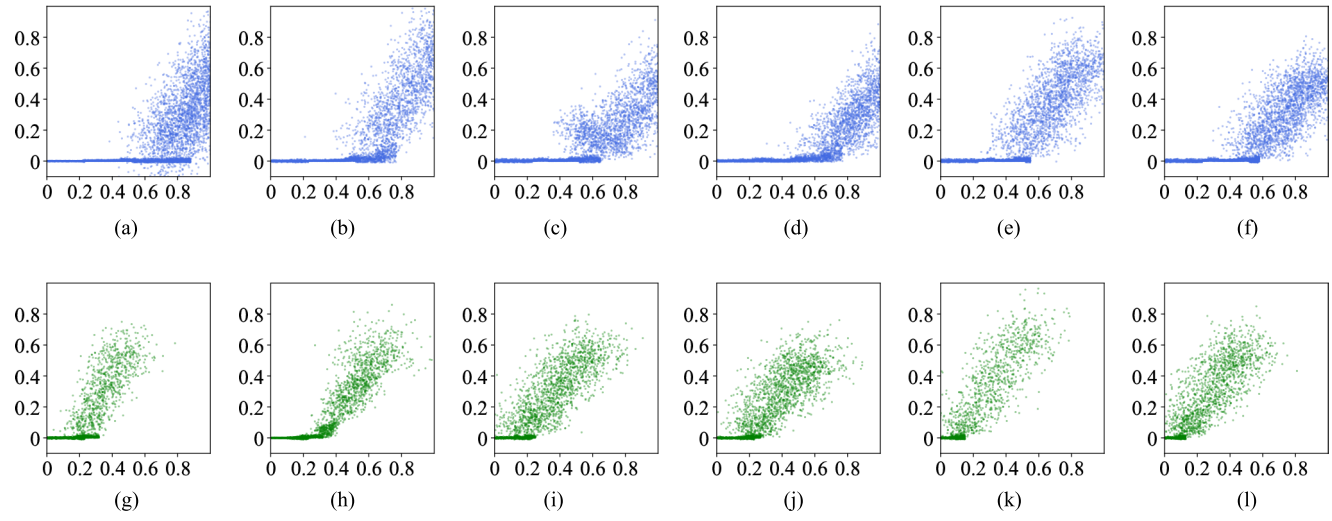

Fig. 5. JS divergence gap (ΔJS) (Y-axis) between attention distribution against $\Delta\alpha$ of attention weight (X-axis) between the most important token i and random selected token r. The results at the topper row in blue color are from the raw attention model, and the results at the corresponding bottom row in green color are from the attention retrofitted by DRAG. (a) DiSAN-SST. (b) DiSAN-SUBJ. (c) IAN-Sem14. (d) IAN-Sem15. (e) Trm-WMT14. (f) Trm-WMT15. (g) DiSAN-SST. (h) DiSAN-SUBJ. (i) IAN-Sem14. (j) IAN-Sem15. (k) Trm-WMT14. (l) Trm-WMT15.

NMT tasks since $\Delta\alpha$ threshold is the lowest (around 0.15). The above analysis shows that DRAG can bring explainability attentions with more intuition.

V. Conclusion

We retrofitted the attention mechanism for sequence modeling in NLP, investigating a principled solution of deep RL for optimizing attention by adding a policy network on top of a baseline attention network for adjusting the attention weights automatically. Without introducing external supervision, our method gives more interpretable attention distribution over four types of attention networks, yielding better task performances. Future work includes how to improve the training efficiency of the RL method, as well as more NLP applications by the proposed method.

Acknowledgment

The authors would like to thank the anonymous reviewers for their detailed comments, which have helped us to improve the quality of this work.

References

[1] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, *arXiv:1409.0473*. [Online]. Available: http://arxiv.org/abs/1409.0473

[2] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," 2015, *arXiv:1508.04025*. [Online]. Available: http://arxiv.org/abs/1508.04025

[3] L. Shang, Z. Lu, and H. Li, "Neural responding machine for short-text conversation," 2015, *arXiv:1503.02364*. [Online]. Available: http://arxiv.org/abs/1503.02364

[4] M. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi, "Bidirectional attention flow for machine comprehension," 2016, *arXiv:1611.01603*. [Online]. Available: http://arxiv.org/abs/1611.01603

[5] Z. Lei, Y. Yang, and M. Yang, "Sentiment lexicon enhanced attention-based LSTM for sentiment classification," in *Proc. Assoc. Advancement Artif. Intell.*, 2018, pp. 8105–8106.

[6] W. Wang, S. J. Pan, D. Dahlmeier, and X. Xiao, "Coupled multi-layer attentions for co-extraction of aspect and opinion terms," in *Proc. Assoc. Advancement Artif. Intell.*, 2017, pp. 3316–3322.

[7] K. Xu *et al.*, "Show, attend and tell: Neural image caption generation with visual attention," 2015, *arXiv:1502.03044*. [Online]. Available: http://arxiv.org/abs/1502.03044

[8] R. Aharoni and Y. Goldberg, "Morphological inflection generation with hard monotonic attention," 2016, *arXiv:1611.01487*. [Online]. Available: http://arxiv.org/abs/1611.01487

[9] J. Serrà, D. Surís, M. Miron, and A. Karatzoglou, "Overcoming catastrophic forgetting with hard attention to the task," 2018, *arXiv:1801.01423*. [Online]. Available: http://arxiv.org/abs/1801.01423

[10] T. Shen, T. Zhou, G. Long, J. Jiang, S. Pan, and C. Zhang, "Disan: Directional self-attention network for RNN/CNN-free language understanding," in *Proc. Association Advancement Artif. Intell.*, 2018, pp. 5446–5455.

[11] A. Vaswani *et al.*, "Attention is all you need," in *Proc. Conf. Workshop Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, 2019, pp. 4171–4186.

[13] Z. Wu, X.-Y. Dai, C. Yin, S. Huang, and J. Chen, "Improving review representations with user attention and product attention for sentiment classification," in *Proc. Association Advancement Artif. Intell.*, 2018, pp. 5989–5996.

[14] T. Alkhouli, G. Bretschner, and H. Ney, "On the alignment problem in multi-head attention-based neural machine translation," 2018, *arXiv:1809.03985*. [Online]. Available: http://arxiv.org/abs/1809.03985

[15] T. Alkhouli and H. Ney, "Biasing attention-based recurrent neural networks using external alignment information," in *Proc. 2nd Conf. Mach. Transl.*, 2017, pp. 108–117.

[16] C. Sen, T. Hartvigsen, B. Yin, X. Kong, and E. Rundensteiner, "Human attention maps for text classification: Do humans and neural networks focus on the same words?" in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 4596–4608.

[17] V. Niculae and M. Blondel, "A regularized framework for sparse and structured neural attention," in *Proc. Conf. Workshop Neural Inf. Process. Syst.*, 2017, pp. 3338–3348.

[18] Q. Zhou, X. Wang, and X. Dong, "Differentiated attentive representation learning for sentence classification," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 4630–4636.

[19] Q. Yin, Y. Zhang, W. Zhang, T. Liu, and W. Yang Wang, "Deep reinforcement learning for chinese zero pronoun resolution," 2018, *arXiv:1806.03711*. [Online]. Available: http://arxiv.org/abs/1806.03711

[20] T. Zhang, M. Huang, and L. Zhao, "Learning structured representation for text classification via reinforcement learning," in *Proc. Assoc. Advancement Artif. Intell.*, 2018, pp. 6053–6060.

[21] D. Ma, S. Li, X. Zhang, and H. Wang, "Interactive attention networks for aspect-level sentiment classification," 2017, *arXiv:1709.00893*. [Online]. Available: http://arxiv.org/abs/1709.00893

[22] B. Zhang, D. Xiong, J. Xie, and J. Su, "Neural machine translation with GRU-gated attention model," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 11, pp. 4688–4698, Nov. 2020.

[23] H. Fei, Y. Ren, and D. Ji, "Implicit objective network for emotion detection," in *Proc. 8th Natural Lang. Process. Chin. Comput.*, 2019, pp. 647–659.
[24] H. Fei, Y. Zhang, Y. Ren, and D. Ji, "Latent emotion memory for multi-label emotion classification," in *Proc. 34th AAAI Conf. Artif. Intell.*, 2020, pp. 7692–7699.
[25] H. Fei, Y. Ren, and D. Ji, "Dispatched attention with multi-task learning for nested mention recognition," *Inf. Sci.*, vol. 513, pp. 241–251, Mar. 2020.
[26] J. Wang *et al.*, "Aspect sentiment classification with both word-level and clause-level attention networks," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 4439–4445.
[27] T. Shen, T. Zhou, G. Long, J. Jiang, S. Wang, and C. Zhang, "Reinforced self-attention network: A hybrid of hard and soft attention for sequence modeling," 2018, *arXiv:1801.10296*. [Online]. Available: http://arxiv.org/abs/1801.10296
[28] M. Ahmed, M. R. Samee, and R. E. Mercer, "You only need attention to traverse trees," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 316–322.
[29] Y. Wang, H.-Y. Lee, and Y.-N. Chen, "Tree transformer: Integrating tree structures into self-attention," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, 2019, pp. 1061–1070.
[30] H. Fei, Y. Ren, and D. Ji, "Retrofitting structure-aware transformer language model for end tasks," 2020, *arXiv:2009.07408*. [Online]. Available: http://arxiv.org/abs/2009.07408
[31] Y. Zhuang and H. Wang, "Token-level dynamic self-attention network for multi-passage reading comprehension," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 2252–2262.
[32] O. Kovaleva, A. Romanov, A. Rogers, and A. Rumshisky, "Revealing the dark secrets of BERT," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, 2019, pp. 4356–4365.
[33] A. Warstadt *et al.*, "Investigating BERT's knowledge of language: Five analysis methods with NPIs," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, 2019, pp. 2870–2880.
[34] S. Wiegreffe and Y. Pinter, "Attention is not not explanation," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process.*, 2019, pp. 11–20.
[35] S. Jain and B. C. Wallace, "Attention is not explanation," in *Proc. Conf. North Amer. Chapter Assoc.*, 2019, pp. 3543–3556.
[36] S. Serrano and N. A. Smith, "Is attention interpretable?" in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 2931–2951.
[37] D. He *et al.*, "Dual learning for machine translation," in *Proc. 13th Annu. Conf. Neural Inf. Process. Syst.*, 2016, pp. 820–828.
[38] L. Yu, W. Zhang, J. Wang, and Y. Yu, "SeqGAN: Sequence generative adversarial nets with policy gradient," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 2852–2858.
[39] M. Fang, Y. Li, and T. Cohn, "Learning how to active learn: A deep reinforcement learning approach," 2017, *arXiv:1708.02383*. [Online]. Available: http://arxiv.org/abs/1708.02383
[40] J. Wu, L. Li, and W. Yang Wang, "Reinforced co-training," 2018, *arXiv:1804.06035*. [Online]. Available: http://arxiv.org/abs/1804.06035
[41] C. Chen, Y. Zhang, and Y. Gao, "Learning how to self-learn: Enhancing self-training using neural reinforcement learning," in *Proc. Int. Conf. Asian Lang. Process. (IALP)*, Nov. 2018, pp. 25–30.
[42] S. R. Indurthi, I. Chung, and S. Kim, "Look harder: A neural machine translation model with hard attention," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 3037–3043.
[43] T. Degris, P. M. Pilarski, and R. S. Sutton, "Model-free reinforcement learning with continuous action in practice," in *Proc. Amer. Control Conf. (ACC)*, Jun. 2012, pp. 2177–2182.
[44] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.
[45] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," in *Proc. Int. Conf. Learn. Represent.*, 2016, pp. 1–14.
[46] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. A. Riedmiller, "Deterministic policy gradient algorithms," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 387–395.
[47] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 229–256, May 1992.
[48] H. Chen, M. Sun, C. Tu, Y. Lin, and Z. Liu, "Neural sentiment classification with user and product attention," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 1650–1659.
[49] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980
[50] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[51] Y. Kim, "Convolutional neural networks for sentence classification," 2014, *arXiv:1408.5882*. [Online]. Available: http://arxiv.org/abs/1408.5882
[52] X. Wu, Y. Cai, Q. Li, J. Xu, and H.-F. Leung, "Combining contextual information by self-attention mechanism in convolutional neural networks for text classification," in *Proc. Int. Conf. Web Inf. Syst. Eng.*, 2018, pp. 453–467.
[53] B. Pang and L. Lee, "Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2005, pp. 115–124.
[54] R. Socher *et al.*, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2013, pp. 1631–1642.
[55] B. Pang and L. Lee, "A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts," in *Proc. 42nd Annu. Meeting Assoc. Comput. Linguistics ACL*, 2004, pp. 271–278.
[56] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Proc. Conf. Workshop Neural Inf. Process. Syst.*, 2015, pp. 649–657.
[57] Y. Wang, M. Huang, X. Zhu, and L. Zhao, "Attention-based LSTM for aspect-level sentiment classification," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 606–615.

Hao Fei received the B.E. degree from Xidian University, Xi'an, China, in 2016, and the M.E. degree from Wuhan University, Wuhan, China, in 2018, where he is currently pursuing the Ph.D. degree with the School of Cyber Science and Engineering.

He has been working on natural language processing and deep learning, especially focusing on language structure parsing, sentiment analysis, and information extraction. He has had his work published at top-tier journals and conferences, including ACL, AAAI, WWW, and EMNLP.

Yue Zhang (Member, IEEE) received the B.E. degree in computer science from Tsinghua University, Beijing, China, in 2003, and the M.S. and Ph.D. degrees from Oxford University, Oxford, U.K., in 2006 and 2009, respectively.

He was an Assistant Professor with the Singapore University of Technology and Design, Singapore, from 2012 to 2018. He is currently an Associate Professor with Westlake University, Hangzhou, China. His research interests include natural language processing, text mining, and machine learning.

Yafeng Ren received the Ph.D. degree from Wuhan University, Wuhan, China, in 2015.

He was a Post-Doctoral Research Fellow with the Singapore University of Technology and Design, Singapore, from 2015 to 2016. He is currently an Associate Professor with the Guangdong University of Foreign Studies, Guangzhou, China. He has been working on natural language processing over the past ten years, and has published 20 related papers in journals and conferences, including AAAI, EMNLP, and COLING. His research interests include opinion mining, biomedical text mining, and bioinformatics.

Donghong Ji received the B.E., M.E., and Ph.D. degrees from the Computer School, Wuhan University, Wuhan, China, in 1988, 1991, and 1995, respectively.

He was a Post-Doctoral Research Fellow with Tsinghua University, Beijing, China, from 1995 to 1998. From 1998 to 2008, he was a Researcher with the Institute for Infocomm Research, Singapore. He is currently a Professor with Wuhan University. His interests include natural language processing, machine learning, and data mining.