Learning to Control the Fine-grained Sentiment for Story Ending Generation

Fuli Luo[1]*, Damai Dai[1,3]*, Pengcheng Yang[1,2], Tianyu Liu[1], Baobao Chang[1,3], Zhifang Sui[1,3], Xu Sun[1,2]
[1]Key Lab of Computational Linguistics, School of EECS, Peking University
[2]Deep Learning Lab, Beijing Institute of Big Data Research, Peking University
[3]Peng Cheng Laboratory, China
{luofuli,daidamai,yang_pc,tianyu0421,chbb,szf,xusun}@pku.edu.cn

Abstract

Automatic story ending generation is an interesting and challenging task in natural language generation. Previous studies are mainly limited to generate coherent, reasonable and diversified story endings, and few works focus on controlling the sentiment of story endings. This paper focuses on generating a story ending which meets the given fine-grained sentiment intensity. There are two major challenges to this task. First is the lack of story corpus which has fine-grained sentiment labels. Second is the difficulty of explicitly controlling sentiment intensity when generating endings. Therefore, we propose a generic and novel framework which consists of a sentiment analyzer and a sentimental generator, respectively addressing the two challenges. The sentiment analyzer adopts a series of methods to acquire sentiment intensities of the story dataset. The sentimental generator introduces the sentiment intensity into decoder via a Gaussian Kernel Layer to control the sentiment of the output. To the best of our knowledge, this is the first endeavor to control the fine-grained sentiment for story ending generation without manually annotating sentiment labels. Experiments show that our proposed framework can generate story endings which are not only more coherent and fluent but also able to meet the given sentiment intensity better.[1]

1 Introduction

Story ending generation aims at completing the plot and concluding a story given a story context. Previous works mainly study on how to generate a coherent, reasonable and diversified story ending (Li et al., 2018; Guan et al., 2018; Xu et al., 2018). However, few of them focus on controllable story ending generation, especially controlling the sentiment for story ending generation. Yao et al. (2018b) is the only work on controlling the sentiment for story ending generation. However, their work needs manually label the story dataset with sentiment labels (happy, sad, unknown), which is time-consuming and labor-intensive. What's more, they only focus on coarse-grained sentiment.

Story context: Sally really loves to play soccer. She joined a team with her friends and she plays everyday. Her coach and her teammates are all really fun. Sally practiced extra hard for her first match.	
Target Sentiment	**Generated Story Endings**
0.1	She still lost the game and was very upset.
0.3	She almost won the game, but eventually lost.
0.5	The game ended with a draw.
0.7	She eventually won the game.
0.9	She won the game and was very proud of her team.

Figure 1: An example of the input story context and output story endings for this task. All of the story endings are coherent with the story context but express different sentiment intensities.

Different from previous work, we propose the task of controlling the sentiment for story ending generation at a fine-grained level, without any human annotation of story dataset[2]. Take Figure 1 as an example, given the same story context, our goal is to generate a story ending that satisfies the given sentiment intensity, where 0 denotes the *most negative* and 1 denotes the *most positive*, following the setting of sentiment intensity on sentiment intensity prediction task (Abdou et al., 2018; Akhtar et al., 2018). To the proposed task, there are two major challenges. First, how to annotate story corpus with sentiment intensities. Second, how to incorporate the fine-grained sentiment control into a generative model.

*Equal Contribution.

[1]Our code and data can be found at https://github.com/luofuli/sentimental-story-ending

[2]Fine-grained sentiment is equivalent to sentiment intensity in this paper.

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 6020–6026
Florence, Italy, July 28 - August 2, 2019.

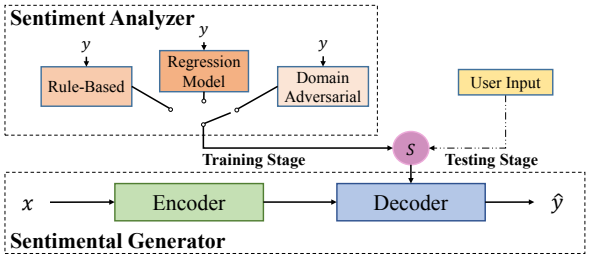

Figure 2: The overview of the proposed framework, which consists of a sentiment analyzer and a sentimental generator. During training, the target sentiment intensity s is computed by the sentiment analyzer. During testing, users can input any sentiment intensity to control the sentiment for story ending generation.

In this work, we propose a framework which consists a sentiment analyzer and a sentimental generator. To address the first challenge, the sentiment analyzer adopts three methods including an unsupervised rule-based method, a regression model, and a domain-adversarial regression model to acquire sentiment intensities of the story training corpus. To address the second challenge, the sentimental generator uses a sentiment intensity controlled sequence-to-sequence model (SIC-Seq2Seq) to generate a story ending which expresses the given sentiment intensity. It introduces an explicit sentiment intensity control variable into the Seq2Seq model via a Gaussian Kernel Layer to guide the generation.

Experiments show the effectiveness and generality of the proposed framework, since it can generate story endings which are not only coherent and fluent but also able to better meet the given sentiment intensity.

2 Proposed Model

2.1 Overview

Here we formulate the task of fine-grained sentiment controllable story ending generation. Given the story context $\boldsymbol{x} = (x_1, \cdots, x_m)$ which consists of m sentences, and the target sentiment intensity s, the goal of this task is to generate a story ending $\boldsymbol{y}$ that is coherent to story context $\boldsymbol{x}$ and expresses the target sentiment intensity s. Note that the sentiment intensity $s \in [0, 1]$.

Although existing datasets for story ending generation can provide paired data $(\boldsymbol{x}, \boldsymbol{y})$, the true sentiment s of $\boldsymbol{y}$ is not observable. To remedy this, the sentiment analyzer $\mathbf{S}$ employs several methods to acquire the sentiment intensity s of $\boldsymbol{y}$. Then the sentimental generator $\mathbf{G}$ takes the story context $\boldsymbol{x}$ and the sentiment of the story ending s as input to generate the story ending $\boldsymbol{y}$. The overview of our proposed framework is presented in Figure 2, which is composed of two modules: a sentiment analyzer $\mathbf{S}$ and a sentimental generator $\mathbf{G}$. The next two sections will show detailed configurations in each module.

2.2 Sentiment Analyzer

The sentiment analyzer $\mathbf{S}$ aims to predicting the sentiment intensity s of the gold story ending $\boldsymbol{y}$ to construct paired data $(\boldsymbol{x}, s; \boldsymbol{y})$. As the first attempt to solve the proposed task, we explore three kinds of sentiment analyzers as follows.

Rule-based (RB): VADER (Hutto and Gilbert, 2014) is an rule-based unsupervised model for sentiment analysis. We use it to extract the sentiment intensity s of $\boldsymbol{y}$ and then scale s to [0, 1].

Regression Model (RM): We first train a linear regression model $\mathbf{R}$ on the Stanford Sentiment Treebank (SST) (Socher et al., 2013) dataset, which is widely-used for sentiment analysis. Then we use $\mathbf{R}$ to acquire the sentiment intensity of $\boldsymbol{y}$.

Domain-Adversarial (DA): In the absence of sentiment annotations for the story dataset, domain adaptation can provide an effective solution since there exists some labeled datasets of a similar task but from a different domain. We use adversarial learning (Ganin and Lempitsky, 2015) to extract a domain-independent feature which not only performs well in the SST sentiment regression task but also misleads the domain discriminator. Finally, we use the adapted regression model to acquire the sentiment intensity s of $\boldsymbol{y}$.

2.3 Sentimental Generator

The sentimental generator $\mathbf{G}$ aims to generate story endings that match the target sentiment intensities s. It consists of an encoder and a decoder equipped with a Gaussian Kernel Layer.

The encoder is to map the input story context $\boldsymbol{x}$ into a compact vector that can capture its essential context features. Specifically, we use a normal bi-directional LSTM as the encoder. All context words x_i are represented by their semantic embeddings $\mathbf{E}$ as the input and we use the concatenation of final forward and backward hidden states as the initial hidden state of the decoder.

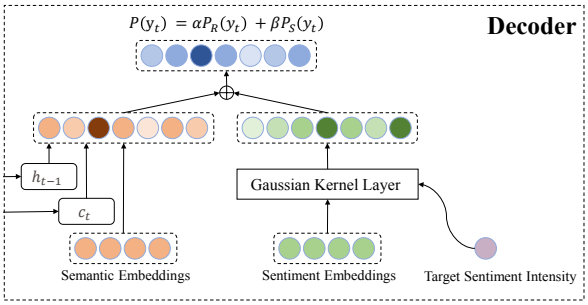

Figure 3: The decoder of the sentimental generator. A Gaussian Kernel Layer is introduced to make use of the target sentiment intensity.

The decoder aims to generate a story ending which accords with the target sentiment intensity s. As shown in Figure 3, the probability of generating a target word P is composed of two probabilities:

$$P(y_t) = \alpha P_R(y_t) + \beta P_S(y_t) \quad (1)$$

where $P_R(y_t)$ denotes the semantic generation probability, $P_S(y_t)$ denotes the sentiment generation probability, α and β are trainable coefficients.

Specifically, $P_R(y_t)$ is defined as follow:

$$P_R(y_t = w) = \mathbf{w}^T(\mathbf{W}_R \cdot \mathbf{h}_{y_t} + \mathbf{b}_R), \quad (2)$$

$$\mathbf{h}_t = \text{LSTM}(y_{t-1}, \mathbf{h}_{t-1}, \mathbf{c}_t) \quad (3)$$

where $\mathbf{w}$ is a one-hot indicator vector of word w, $\mathbf{W}_R$ and $\mathbf{b}_R$ are trainable parameters, $\mathbf{h}_t$ is the t-th hidden state of the LSTM decoder with attention mechanism (Luong et al., 2015).

$P_S(y_t)$ measures the generation probability of the target word given the target sentiment intensity s. For all words, beyond their semantic embeddings, they also have sentiment embeddings $\mathbf{U}$. The sentiment embeddings of words reflect their sentiment properties. A Gaussian Kernel Layer (Luong et al., 2015; Zhang et al., 2018) is used to encourage words with sentiment intensity near to target sentiment s, and $P_S(y_t)$ is defined as follow:

$$P_S(y_t = w) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(\Phi_S(\mathbf{U}\mathbf{w}) - s)^2}{2\sigma^2}\right) \quad (4)$$

$$\Phi_S(\mathbf{U}, \mathbf{w}) = \text{sigmoid}(\mathbf{w}^T(\mathbf{U} \cdot \mathbf{W}_U + \mathbf{b}_U)) \quad (5)$$

where σ^2 is the variance, Φ_S maps the sentiment embedding into a real value, the target sentiment intensity s is the mean of the Gaussian distribution, $\mathbf{W}_U$ and $\mathbf{b}_U$ are trainable parameters.

3 Experiment

3.1 Dataset

We choose the widely-used ROCStories corpus (Mostafazadeh et al., 2016) which consists of 100k five-sentence stories. We split the data into a training set with 93,126 stories, a validation set with 5,173 stories and a test set with 5,175 stories.

3.2 Baselines

Since there is no direct related work of this task, we design an intuitive pipeline (*generate-and-modify*) as baseline. It first *generates* a story ending using a general sequence-to-sequence model with attention (Luong et al., 2015), and then *modifies* the sentiment of the story ending towards the target sentiment intensity via a fine-grained sentiment modification method (Liao et al., 2018). We call this baseline **Seq2Seq + SentiMod**.

3.3 Experiment Settings

We tune hyper-parameters on the validation set. For the RM and DA sentiment analyzer, we implement the encoder as a 3-layer bidirectional LSTM with a hidden size of 512. We implement the regression module as a MLP with 1 hidden layer of size 32. For domain adaption, we implement a domain discriminator as a MLP with 1 hidden layer of size 32. A Gradient Reversal Layer is added into the domain discriminator. For the sentimental generator, both the semantic and sentiment embeddings are 256 dimensions and randomly initialized. We implement both encoder and decoder as 1-layer bidirectional LSTM with a hidden size of 512. The variance σ^2 of Gaussian Kernel Layer is set as 1. The batch size is 32 and the dropout (Srivastava et al., 2014) is 0.5. We use the Adam optimizer (Kingma and Ba, 2014) with an initial learning rate of 0.0003.

3.4 Evaluation Metrics

For the proposed task, there are no existing accepted metrics. We propose both automatic evaluation and human evaluation for this task.

3.4.1 Automatic Evaluation

Sentiment Consistency: We propose the pairwise sentiment consistency (SentiCons) to evaluate the consistency of two lists of sentiment intensities. For two lists A and B with the same length,

Model	H-M SentiCons
Rule-Based (RB)	**0.936**
Regression Model (RM)	0.846
Domain Adversarial (DA)	0.747

Table 1: Automatic evaluation of sentiment analyzers.

$\text{SentiCons}(A, B)$ is calculated by

$$\frac{\sum_{1\le i<j\le n} \mathbb{I}_{(A_i\le A_j \wedge B_i\le B_j)\vee(A_i\ge A_j \wedge B_i\ge B_j)}}{C_n^2}, \quad (6)$$

where n is the length of the list and $\mathbb{I}$ is the indicator function. To evaluate the performance of sentiment analyzer, we calculate SentiCons of human-annotated sentiment intensities and model-predicted sentiment intensities of gold story endings in the test set (**H-M SentiCons**). To evaluate the performance of sentimental generator, for each story context in the test set, we generate five story endings with five target sentiment intensity ranging from [0, 1]. Then we calculate SentiCons of input target sentiment intensities and sentiment intensities of the outputs predicted by the best sentiment analyzer (**I-O SentiCons**).

BLEU: For each story in the test set, we take the context $\boldsymbol{x}$ and the human-annotated sentiment intensity s of the gold story ending $\boldsymbol{y}$ as input. The corresponding output is $\hat{\boldsymbol{y}}$. Then we calculate the BLEU (Papineni et al., 2002) score of $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$ as the overall quality of the generated story endings.

3.4.2 Human Evaluation

We hire two evaluators who are skilled in English to evaluate the generated story endings. For each story in the test set, we distribute the story context, five target sentiment intensities and corresponding generated story endings to the evaluators. Evaluators are required to score the generated endings from 1 to 5 in terms of three criteria: **Coherency**, **Fluency** and **Sentiment**. Coherency measures whether the endings are coherent with the context. Fluency measures whether the endings are fluent. Sentiment measures how much the endings express the target sentiment intensities.

3.5 Evaluation Results

Table 1 shows the automatic evaluation results of three sentiment analyzers. We find that: (1) The rule-based method RB performs the best. This accords with the fact that story endings in the ROCStories corpus are simple and have relatively obvious emotional words. (2) DA can not improve the performance of sentiment analysis in our task compared to RM. We hypothesize that is because the domains of labeled SST corpus and ROCStories corpus differ too much that affects the performance of domain adaptation.

Model	BLEU-1	BLEU-2	I-O SentiCons
Seq2Seq + SentiMod	10.7	3.2	0.788
SIC-Seq2Seq + RB	19.3	6.3	**0.879**
SIC-Seq2Seq + RM	19.5	6.2	0.830
SIC-Seq2Seq + DA	**19.8**	**6.7**	0.794

Table 2: Automatic evaluation of generation models.

Model	Coherency	Fluency	Sentiment
Seq2Seq + SentiMod	1.50	2.50	3.68
SIC-Seq2Seq + RB	**2.65**	**4.75**	**4.09**
SIC-Seq2Seq + RM	2.15	4.60	3.65
SIC-Seq2Seq + DA	2.20	4.50	3.71

Table 3: Human evaluation of generation models.

The automatic and human evaluation results of four generation models are shown in Table 2 and Table 3 respectively. We have the following observations: (1) Three models based on our proposed framework do not have obvious performance difference in terms of BLEU, Coherency, and Fluency. Meanwhile, all of them can largely outperform the Seq2Seq+SentiMod baseline which does not follow our framework. Thus it shows the effectiveness of the proposed framework. (2) H-M SentiCons which measures the performance of sentiment analyzer is marginally consistent with the I-O SentiCons and Sentiment which measure the performance of sentimental generator. This accords with our expectations because the sentimental generator takes the sentiment intensity predicted by the sentiment analyzer as the input signal for controlling the sentiment of the output.

From a comprehensive perspective, our framework can better control the sentiment while guaranteeing the coherency and fluency.

4 Case Study

We provide an example of story ending generation with five different target sentiment intensities in Table 4. This demonstrates that our proposed framework can generate more fluent and coherent story endings than the Seq2Seq + SentiMod baseline which does not follow our framework. More importantly, at the same time, our framework has better control over the sentiment tenden-

Story Context	Madison really wanted to buy a new car. She applied to work at different restaurants around town. One day a local restaurant hired her to be their new waitress! Molly worked very hard as a waitress and earned a lot of tips.
Outputs	**Seq2Seq + SentiMod**
$\mathbf{s = 0.1}$	Dates sangria and drinks went loved the drinks!
$\mathbf{s = 0.3}$	Madison was never in once some showed up.
$\mathbf{s = 0.5}$	Madison's finally cut and delicious wine.
$\mathbf{s = 0.7}$	Madison was happy so new great hospital!
$\mathbf{s = 0.9}$	Tom and satisfied big meal and sweet!
Outputs	**SIC-Seq2Seq + RB**
$\mathbf{s = 0.1}$	Madison got in trouble for not buying the car again.
$\mathbf{s = 0.3}$	Madison was so embarrassed that she threw her car out.
$\mathbf{s = 0.5}$	Madison was able to buy her car.
$\mathbf{s = 0.7}$	Madison was so excited to be able to buy her car!
$\mathbf{s = 0.9}$	Madison was happy to have a new car and be happy with her new car!

Table 4: Example outputs with five different target sentiment intensities s ranging from 0 to 1. The generated story endings of the baseline (Seq2Seq + SentiMod) are shown at the top. The generated story endings of the best proposed model (SIC-Seq2Seq + RB) are shown at the bottom.

cies of generated story endings, e.g. "in trouble" $\rightarrow$ "embarrassed" $\rightarrow$ "able to" $\rightarrow$ "excited" $\rightarrow$ "happy" and "new car".

5 Related Work

Story generation Automatic story generation has attracted interest over the past few years. Recently, many approaches are proposed to generate a better story in terms of coherence (Jain et al., 2017; Xu et al., 2018), rationality (Li et al., 2018), topic-consistence (Yao et al., 2018a). However, most of story generation methods lack the ability to receive guidance from users to achieve a specific goal. There are only a few works focus on the controllability of story generation, especially on sentiment. Tambwekar et al. (2018) introduces a policy gradient learning approach to ensure that the model ends with a specific type of event given in advance. Yao et al. (2018b) uses manually annotated story data to control the ending valence and storyline of story generation. Different from them, our proposed framework can acquire distant sentiment labels without the dependence on the human annotations.

Sentimental Text Generation Generating sentimental and emotional texts is a key step towards building intelligent and controllable natural language generation systems. To date several works of dialogue generation (Zhou et al., 2018; Huang et al., 2018; Zhou and Wang, 2018) and text sentiment transfer task (Li et al.; Luo et al., 2019) have studied on generating emotional or sentimental text. They always pre-define a binary sentiment label (positive/negative) or a small limited set of emotions, such as "anger", "love". Different from them, controlling the fine-grained sentiment (a numeric value) for story ending generation is not limited to several emotional labels, thus we can not embed each sentiment label into a separate vector as usual. Therefore, we propose to introduce the numeric sentiment value via a Gaussian Kernel Layer.

6 Conclusion and Future Work

In this paper, we make the first endeavor to control the fine-grained sentiment for story ending generation. The proposed framework is generic and novel, and does not need any human annotation of story dataset. Experiments show the effectiveness of the proposed framework to control the sentiment intensity on both automatic evaluation and human evaluation. Future work can combine the analyzer and generator via joint training, hopefully to achieve better results.

Acknowledgments

This paper is supported by NSFC project 61772040 and 61876004. The contact authors are Baobao Chang and Zhifang Sui.

References

Mostafa Abdou, Artur Kulmizev, and Joan Ginés i Ametllé. 2018. Affecthor at semeval-2018 task 1: A cross-linguistic approach to sentiment intensity quantification in tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT*.

Md. Shad Akhtar, Deepanway Ghosal, Asif Ekbal, and Pushpak Bhattacharyya. 2018. A multi-task ensemble framework for emotion, sentiment and intensity prediction. In *arXiv preprint arXiv:1808.01216*.

Yaroslav Ganin and Victor S. Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *ICML*.

Jian Guan, Yansen Wang, and Minlie Huang. 2018. Story ending generation with incremental encoding and commonsense knowledge. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, AAAI-18*.

Chenyang Huang, Osmar Zaiane, Amine Trabelsi, and Nouha Dziri. 2018. Automatic dialogue generation with expressed emotions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*.

Clayton J Hutto and Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Eighth international AAAI conference on weblogs and social media*.

Parag Jain, Priyanka Agrawal, Abhijit Mishra, Mohak Sukhwani, Anirban Laha, and Karthik Sankaranarayanan. 2017. Story generation from sequence of independent short descriptions. *SIGKDD*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Juncen Li, Robin Jia, He He, and Percy Liang. Delete, retrieve, generate: a simple approach to sentiment and style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018*.

Zhongyang Li, Xiao Ding, and Ting Liu. 2018. Generating reasonable and diversified story ending using sequence to sequence model with adversarial training. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018*.

Yi Liao, Lidong Bing, Piji Li, Shuming Shi, Wai Lam, and Tong Zhang. 2018. Quase: Sequence editing under quantifiable guidance. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Fuli Luo, Peng Li, Jie Zhou, Pengcheng Yang, Baobao Chang, Zhifang Sui, and Xu Sun. 2019. A dual reinforcement learning framework for unsupervised text style transfer. *CoRR*, abs/1905.10060.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and evaluation framework for deeper understanding of commonsense stories. *arXiv preprint arXiv:1604.01696*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*.

Pradyumna Tambwekar, Murtaza Dhuliawala, Animesh Mehta, Lara J. Martin, Brent Harrison, and Mark O. Riedl. 2018. Controllable neural story generation via reinforcement learning. *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, AAAI-18*.

Jingjing Xu, Xuancheng Ren, Yi Zhang, Qi Zeng, Xiaoyan Cai, and Xu Sun. 2018. A skeleton-based model for promoting coherence among sentences in narrative story generation. In *Proceedings of EMNLP*.

Lili Yao, Nanyun Peng, Ralph M. Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2018a. Plan-and-write: Towards better automatic storytelling. *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, AAAI-18*.

Lili Yao, Nanyun Peng, Ralph M. Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2018b. Towards controllable story generation. *NAACL Workshop*.

Ruqing Zhang, Jiafeng Guo, Yixing Fan, Yanyan Lan, Jun Xu, and Xueqi Cheng. 2018. Learning to control the specificity in neural response generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Hao Zhou, Minlie Huang, Tianyang Zhang, Xiaoyan Zhu, and Bing Liu. 2018. Emotional chatting machine: Emotional conversation generation with internal and external memory. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, AAAI-18*.

Xianda Zhou and William Yang Wang. 2018. Mojitalk: Generating emotional responses at scale. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018*.