Research Collection School Of Computing and Information Systems

School of Computing and Information Systems

11-2019

Aspect and opinion aware abstractive review summarization with reinforced hard typed decoder

Yufei TIAN
Tsinghua University

Jianfei YU
Singapore Management University, jfyu@smu.edu.sg

Jing JIANG
Singapore Management University, jingjiang@smu.edu.sg

Follow this and additional works at: https://ink.library.smu.edu.sg/sis_research

Part of the Databases and Information Systems Commons, and the Numerical Analysis and Scientific Computing Commons

Aspect and Opinion Aware Abstractive Review Summarization with Reinforced Hard Typed Decoder

Yufei Tian
Tsinghua University
tyf16@mails.tsinghua.edu.cn

Jianfei Yu*
Singapore Management University
jfyu@smu.edu.sg

Jing Jiang
Singapore Management University
jingjiang@smu.edu.sg

ABSTRACT

In this paper, we study abstractive review summarization. Observing that review summaries often consist of aspect words, opinion words and context words, we propose a two-stage reinforcement learning approach, which first predicts the output word type from the three types, and then leverages the predicted word type to generate the final word distribution. Experimental results on two Amazon product review datasets demonstrate that our method can consistently outperform several strong baseline approaches based on ROUGE scores.

KEYWORDS

Natural Language Processing; Review Summarization; Text Generation; Neural Networks

1 INTRODUCTION

Reviews posted on e-commerce platforms by online shoppers are valuable resources for businesses to improve product quality and keep track of customer preferences. However, given the large amount of product reviews in real scenarios, it is impractical to manually read through each review, especially when they are lengthy and have low readability. Therefore, it is crucial to design a robust model to automatically generate concise and readable summaries for product reviews, which is often referred to as *review summarization* [3].

In the literature, existing approaches to *review summarization* generally belong to two groups: extractive summarization and abstractive summarization. The former line of work focuses on selecting several informative sentences or phrases from a set of reviews of a product [1, 12]. The latter centers on generating a short but meaningful summary for either a single review of a product [15, 16] or multiple reviews of a product [3, 4]. In this paper, following Yang

---

*Corresponding author.

E.g. 1
Product Review: *This is a good little case and fits my laptop hard drive well. You might want to measure your hard drive first and check it against the measurement in the description!*
ABS : *nice small case*
PGNet: *good case, quality is clear*
Our Model: *great case for laptop hard drive*
Human: *good protection for a portable hard drive*

Figure 1: Current models tend to output general and less meaningful summaries.

et al. [15], we aim to develop an effective model that can generate a concise summary for a single input review.

As with any abstractive summarization task, current representative encoder-decoder frameworks including the well-known sequence-to-sequence (*Seq2Seq*) model [8] and the advanced pointer-generator network (*PGNet*) [5, 10] are natural to be adopted. More recently, observing that some under-represented aspect and opinion words tend to be ignored by *Seq2Seq* and *PGNet*, Yang et al. [15] proposed a multi-factor attention network based on *PGNet*, which forces their model to focus more on aspect and opinion words in their modified attention mechanism. However, since all these approaches assume to generate words from the same vocabulary at each decoding step, they tend to produce generic summaries with high-frequency phrases as in Fig. 1, which often fail to include those less frequent aspect or opinion words that are also essential to review summaries. As illustrated in Fig.1, an informative review summary written by human should be a natural composition of *aspect words*, *opinion words*, and *context words*, where *aspect words* and *opinion words* indicate the product information and users' opinions respectively, and *context words* are used to make the summary coherent.

Motivated by this, we borrow the idea from a state-of-the-art dialogue question generation model [13], and aim to explicitly control word types when generating the review summary. Specifically, we first classify all the vocabulary words into three types: *aspect words*, *opinion words*, and *context words*. Based on this, we propose a two-stage Reinforced Hard Typed Decoder (*RHTD*), which first explicitly predicts the output word type, followed by generating the final word distribution based on the predicted type at each decoding position. Due to the discrete choice of word types, the gradient over the first stage becomes non-differentiable. To jointly optimize the two stages, instead of simply following Wang et al. [13] to use Gumbel-Softmax [6] as a differentiable approximation, we adopt a widely used policy gradient algorithm REINFORCE [14] to assign an explicit feedback reward to the predicted word type.

We carry out experiments on two Amazon product review datasets. Automatic evaluation based on ROUGE scores [7] demonstrates that *RHTD* outperforms several highly competitive baseline approaches. Further analysis verifies that *RHTD* can indeed produce more informative summaries.

Rules	Relations (PoS Tags)	Examples
R1	AP (**NN**) $\xrightarrow{nn}$ AP (**NN**)	The Mac $\xrightarrow{nn}$ OS is excellent.
R2	OP (**JJ**) $\xrightarrow{conj}$ OP (**JJ**)	It is light and $\overset{conj}{\frown}$ portable.
R3	AP (**NN**) $\xrightarrow{nsubj}$ OP (**JJ**)	The speed is $\overset{nsubj}{\frown}$ incredible.
R4	OP (**JJ**) $\xrightarrow{amod}$ AP(**NN**)	iPhone has great $\xrightarrow{amod}$ design.

Table 1: Rules for extracting aspect words (AP) and opinion words (OP). NN and JJ respectively denote two sets of Part-of-Speech Tags (PoS Tags), i.e., {NN, NNS} and {JJ, JJS and JJR}. *conj, nn, amod,* and *nsubj* are dependency relations.

2 METHODOLOGY

In this section, we first formulate our task. We then introduce our aspect and opinion words extraction approach, and review Pointer-Generator Network. Finally, we describe a modified Hard Typed Decoder model, followed by our reinforcement learning method.

2.1 Task Definition

We are given a set of user product reviews $\mathcal{D}$, and each product review $X \in \mathcal{D}$ is associated with a short summary Y. The task of abstractive review summarization can be formalized as follows: given a source product review with m words $X = x_1, x_2, ..., x_m$, the system should generate a concise and informative target summary with n words $Y^* = y_1, y_2, ..., y_n$ that captures the salient points, formally as $Y^* = argmax_Y P(Y|X)$.

2.2 Aspect and Opinion Words Extraction

As mentioned before, we assume that a well-informed review summary should consist of three types of words: aspect, opinion, and context words. However, due to the labor intensive nature of human annotation, it is almost impossible to manually collect all the aspect and opinion words. Therefore, we employ the well-known unsupervised extraction method, Double Propagation [9], to automatically extract all the aspect and opinion words in each domain.

Following Qiu et al. [9], we leverage four syntactic rules in Table 1 to identify potential aspect and opinion words. Specifically, we first utilize a sentiment lexicon[1] to extract all the opinion words occurring in source product reviews of $\mathcal{D}$, and then expand the opinion word list based on *R2* in Table 1. Given the extracted opinion words, we further use *R3* and *R4* to extract aspect words from $\mathcal{D}$. For example, in *R3*, since *incredible* is detected as an opinion word and the subject of *incredible* is usually aspect words, we can employ this rule to detect that its subject *speed* is an aspect word. Next, the aspect word list is also expanded based on *R1*. Finally, we can make use of the above four rules to iteratively expand the aspect and opinion word lists. Based on the identified aspect and opinion words, let us use A, O, and C to respectively denote the three word types {aspects, opinions, context words}, V the whole vocabulary.

2.3 Pointer-Generator Network (PGNet)

Since *PGNet* is essentially a combination of *Seq2Seq* [8] and a pointer network [11] and has been shown to outperform *Seq2Seq* in many

[1]https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html#lexicon
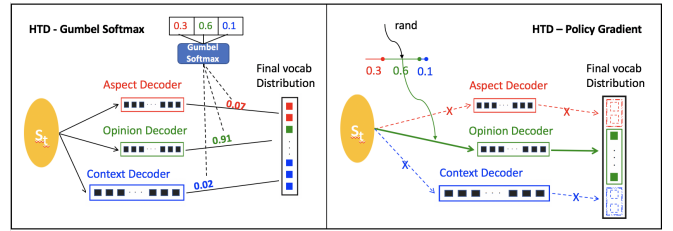

Figure 2: Overview of Hard Typed Decoder (left) and Our Reinforced Hard Typed Decoder (right).

generation tasks [5, 10], we adopt it as our base model. First, let us introduce the necessary notation for *Seq2Seq*. We use s_t to denote the decoder state at time step t, a_k^t the attention weight over each encoder hidden state h_k, and $h_t^\star$ the weighted sum of encoder hidden states. To generate the word distribution over V at time step t, $h_t^\star$ and s_t are concatenated together by feeding them to a linear function:

$$P_{\text{vocab}}(w_t) = \text{softmax}(W^T[s_t, h_t^\star] + b), \tag{1}$$

where W and b are learnable parameters.

In *PGNet*, a generation probability $p_{\text{gen}} \in [0, 1]$ is introduced to control whether to generate a word from V or copy words from the input sequence X via pointer network at time step t:

$$p_{\text{gen}} = \sigma(w_h^T h_t^\star + w_s^T s_t + w_x^T x_t + b_{ptr}) \tag{2}$$

where w_h, w_s, w_x and b_{ptr} are parameters to be learned. The final probability distribution over the extended vocabulary is:

$$P_{\text{vocab}}(w_t) = p_{\text{gen}}P_{\text{vocab}}(w_t) + (1-p_{\text{gen}}) \sum_{k:w_k=w_t} a_k^t, \tag{3}$$

where the first and second terms are respectively referring to the generation distribution in Eq. (1) and the distribution over the input sequence X by sampling from the attention distribution a^t for the encoder hidden states.

2.4 Proposed Approach

Recall that to help our model pay more attention to aspect and opinion words to generate more informative summary, we propose to first explicitly control the type of the output word, followed by generating the word distribution based on the predicted type at each decoding step. We formulate this process as follows:

$$c_t^* = \arg\max_{c_i} P(tp_{w_t} = c_i|w_{<t}, X), i \in \{0, 1, 2\}, \tag{4}$$

$$P_{\text{vocab}(c_t^*)}(w_t) = P(w_t|tp_{w_t} = c_t^*), \tag{5}$$

where c_i is one of the three word types $\{A, O, C\}$ and tp_{w_t} denotes the word type at time step t. Note that we split all the words in V and X into aspect, opinion, and context words respectively, and $P_{\text{vocab}(c_t^*)}(w_t)$ is a type-specific word distribution.

However, the choice of word type (i.e., argmax) is discrete and non-differentiable. Therefore, we propose the following two solutions to tackle this problem.

2.4.1 Hard Typed Decoder

As shown in Fig. 2, we first borrow the idea from the Hard Typed Decoder (*HTD*) proposed by Wang et al. [13], and use Gumbel-Softmax (GS) [6] as a differentiable surrogate. As the *HTD* model employed by Wang et al. [13] is simply based on *Seq2Seq*, here we

adapt it to *PGNet* with some modifications. Specifically, to approximate $P_{\text{vocab}}(w_t)$, we introduce:

$$P'_{\text{vocab}}(w_t) = P(w_t|tp_{w_t} = c_i) \cdot \boldsymbol{m}(w_t), i \in \{0, 1, 2\}, \quad (6)$$

where

$$\boldsymbol{m}(w_t) = \mathbf{GS}\big(P(tp_{w_t} = c_i|w_{<t}, X)\big), i \in \{0, 1, 2\}, \quad (7)$$

$$\mathbf{GS}(X_i) = \frac{e^{(log(X_i)+g_i)/\tau}}{\sum_{j=0}^{2} e^{(log(X_j)+g_j)/\tau}}. \quad (8)$$

Note that g_i are i.i.d samples drawn from the Gumble distribution, and τ is a hyper-parameter to control the smoothness of the distribution. The closer constant τ is to 0, the similar Eq.(6) is to argmax. We set τ to 1 to make GS smoother than argmax but can also exhibit the **hard** characteristics.

Similarly, we modify the copy distribution a_k^t over the input sequence X to be $\beta_k^t = a_k^t \cdot \boldsymbol{m}(w_t)$, and the final generation probability distribution can be calculated as follows:

$$P(w_t) = p_{\text{gen}} P'_{\text{vocab}}(w_t) + (1 - p_{\text{gen}}) \sum_{k:w_k=w_t} \beta_k^t. \quad (9)$$

Finally, the loss function for *HTD* is essentially a combination of copy, generation, and type loss:

$$\mathcal{J}_h = \sum_t -\big(\log P(w_t) + \lambda \log P(tp_{w_t}|w_{<t}, X)\big) \quad (10)$$

where λ is a hyper-parameter.

2.4.2 Reinforced Hard Typed Decoder

Motivation: Although the above *HTD* model can eliminate the non-differentiable gradient issue, it mainly suffers from the following problem. Since the modified GS distribution is much sharper than Softmax, it may lead to severe error propagation to the following word generation process if the original Softmax distribution significantly deviates from the reference type distribution. Inspired by this, we propose to jointly train the two stages with REINFORCE algorithm [14], which can largely eliminate the error propagation issue by sampling a word type based on the original Softmax distribution.

Specifically, we first initialize all the model parameters with a well-trained *HTD* model. Given an input review X and its generated word $w_{<t}$ before time step t, we first calculate the type distribution $P(tp_{w_t}|w_{<t}, X)$ in Eq.(4), followed by sampling a word type $c(\bar{w}_t)$ at time step t, where $\bar{w}_t$ denotes the output word from the second stage. Then, the gradient for the second stage is as below:

$$\nabla_{\phi_2} \mathcal{J}_2(\phi_2) = \nabla_{\phi_2} - \log P(w_t) \quad (11)$$

Next, the rewards for training the first stage is calculated as follows:

$$v_t = \begin{cases} 0.3, & c(\bar{w}_t) \neq c(w_t^*) \\ 1.0, & c(\bar{w}_t) = c(w_t^*) \end{cases} \quad (12)$$

where $c(w_t^*)$ is the reference word type. The gradient for the first stage is then computed based the policy gradient theorem [14]:

$$\nabla_{\phi_1} \mathcal{J}_1(\phi_1) = \mathbb{E}[v_t \cdot \nabla_{\phi_1}(-\log P(tp_{w_t}|w_{<t}, X))]$$

where the sampling approach is used to estimate the expected reward. We repeat the above iterative training process until convergence.

3 EXPERIMENTS

3.1 Experiment Settings

Datasets: We evaluate our model on Amazon reviews dataset[2], and select two domains from the raw dataset to conduct our experiments, which are `Healthcare` and `Electronics`.

Pre-processing Details: For both datasets, we filter out review-summary pairs that are too long/short to expedite training and testing, and obtained 48,495 and 187,143 valid review-summary pairs. We then randomly split them into training (70%), development (10%) and test sets (20%). Next, as introduced in Section 2.2, we applied Double Propagation method [9] on the training set to extract 3,104 aspect words and 2,118 opinion words for `Healthcare` domain. The number for `Electronics` domain is 14,305 and 11,232.

Parameter Settings: For all the experiments, we set the word embedding size e to be 128, and initialize the word embedding matrix $\mathbf{E}$ using pre-trained word embeddings based on Glove[3], which will be fixed during the training process. The hidden dimension d and the number of LSTM layers in both datasets are set to be 128 and 1. During training, we adopt Adagrad [2] with learning rate 0.05. Note that we initialize the parameters in *RHTD* with a pre-trained *HTD* model.

Evaluation Metrics: Following many previous studies on abstractive summarization, we choose ROUGE-1, 2, L [7] to automatically quantify how well a model fits the data.

3.2 Main Results

Model	Healthcare			Electronics		
	R-1	R-2	R-L	R-1	R-2	R-L
Seq2Seq	19.33	9.31	18.25	22.71	11.49	21.14
PGNet	25.70	12.36	24.02	28.29	14.35	26.38
STD	25.54	12.42	23.92	27.58	14.06	26.09
HTD	27.59	12.74	25.64	28.63	14.70	27.21
RHTD	**28.67**	**13.26**	**26.58**	**31.97**	**15.23**	**30.11**

Table 2: ROUGE F1 scores of different methods on two domains' test sets for abstractive review summarization.

In this subsection, we compare our proposed *RHTD* with the following four strong baseline approaches: 1). **Seq2Seq**: the standard encoder-decoder RNNs coupled with attention mechanism proposed by Nallapati et al. [8]; 2). **PGNet**: the Pointer-Generator Network proposed by See et al. [10] that can both copy words from the source text via pointer network, and produce novel words via the generator; 3). **STD**: the Soft Typed Decoder proposed by Wang et al. [13], which also incorporates three separate decoders for each word type, but simply forces the three decoders to share the whole vocabulary, and employs the weighted sum of the three word distribution as the final word distribution. Note that we adapt the original *STD* model to *PGNet*. 4). **HTD**: our modified Hard Typed Decoder based upon *PGNet*, as introduced in Section 2.4.1; 5). **RHTD**: our full model with reinforcement learning, as introduced in Section 2.4.2.

Based on the ROUGE scores reported in Table 2, it is easy to observe that the performance of *Seq2Seq* is relatively limited. *PGNet* and *STD* can bring significant improvements over *Seq2Seq* perhaps
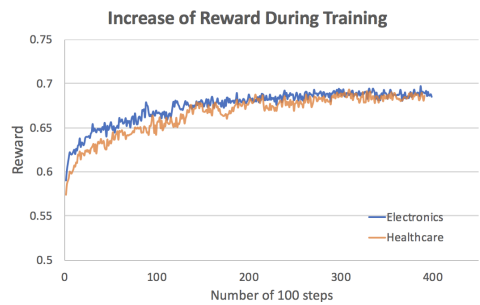
---

[2]http://jmcauley.ucsd.edu/data/amazon/.
[3]https://nlp.stanford.edu/projects/glove/.

Figure 3: Learning Curve of *RHTD*

Figure 4: Generated Summaries on E.g. 2

Figure 5: Generated Summaries on E.g. 3

due to the incorporation of copy mechanism.Moreover, by explicitly incorporating three type-specific decoders, our *HTD* and *RHTD* models can further boost the performance of *PGNet* and *STD* with a large margin. Finally, we can find that *RHTD* consistently outperforms all its competitors, and obtains 3.67% and 10.66% performance gains over the second best model for Healthcare and Electronics, respectively. The higher performance on Electronics might result from a larger size of training data.

3.3 Further Analysis

Visualization of Rewards Increase: To show the advantages of our *RHTD* model, we further plot the reward factor v_t (defined in Eq.12) for both datasets in Fig. 3. Compared to *HTD* (i.e., at step 0), *RHTD* gradually increased the reward by 0.1, meaning that it is better at predicting the right word types (i.e., reference types).

Case Study: To have a better understanding of the advantage of our model, we select two representative examples in Fig. 4 and Fig. 5 to perform human analysis.

First, we can easily observe that all typed decoders indeed generate more informative responses. As we can see, *Seq2Seq* tends to generate low-quality summaries like *'great product'* or *'hair and hair'*, and *PGNet* also outputs universal phrases like *'this is a great product'*, or *'I take it'*. On the contrary, all three typed decoders output aspect and opinion words like *'groomer'*, *'pet'*, *'heartburn'*, *'supplement'*, *'medication'* and *'stomach'*, making the summaries more instructive to potential buyers.

Second, *RHTD* is better at extracting the most salient points from input. In Fig. 4, we can see from the original review that this groomer is shared by the purchaser's family and dog. While *HTD* only captures partial information and concludes *'awesome for pet groomers'*, *RHTD* gets a more holistic view by outputting *'satisfying groomer for us and dog'*, which is closer to human-uttered summary. On the other hand, the review in Fig. 5 covers 4 points regarding the antacid tablet: 1). it is used for stomachache; 2). it works fast; 3). it has no chalky taste; and 4). it contains calcium. *Seq2Seq* and *PGNet* generate vague summaries that miss the point. *STD* is better than the previous two by covering the tablet's usage and effectiveness, but unfortunately copies the wrong word *'heartburn'*. *HTD* adds that the tablets are *'tasty'*, but mentions nothing about what the medicine is used for. Comparatively, *RHTD* is most comprehensive by both stating that it is *'effective'* and that *'flavor is good'*. Moreover, *RHTD* is the only non-human model that correctly states the usage: *'medication for stomach'*. In summary, *RHTD* takes a more comprehensive look at longer product reviews.

4 CONCLUSION

We presented a two-stage reinforcement learning approach for abstractive review summarization, which first predicts the output word type, and then generates the final word distribution based on the predicted type. Evaluations on two Amazon product review datasets show the effectiveness of our method. Finally, we believe that the idea of typed decoders can be applied to a variety of NLP tasks.

ACKNOWLEDGMENTS

REFERENCES

[1] Stefanos Angelidis and Mirella Lapata. 2018. Summarizing Opinions: Aspect Extraction Meets Sentiment Prediction and They Are Both Weakly Supervised. In *Proceedings of EMNLP*.
[2] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12, Jul (2011), 2121–2159.
[3] Kavita Ganesan, ChengXiang Zhai, and Jiawei Han. 2010. Opinosis: A graph based approach to abstractive summarization of highly redundant opinions. In *Proceedings of COLING*.
[4] Shima Gerani, Yashar Mehdad, Giuseppe Carenini, Raymond T Ng, and Bita Nejat. 2014. Abstractive summarization of product reviews using discourse structure. In *Proceedings of EMNLP*. 1602–1613.
[5] J Gu, Z Lu, H Li, and VOK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of ACL*.
[6] Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144* (2016).
[7] Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*.
[8] Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023* (2016).
[9] Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2011. Opinion word expansion and target extraction through double propagation. *Computational linguistics* 37, 1 (2011), 9–27.
[10] Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of ACL*.
[11] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Proceedings of NIPS*.
[12] Lu Wang and Wang Ling. 2016. Neural Network-Based Abstract Generation for Opinions and Arguments. In *Proceedings of NAACL-HLT*.
[13] Yansen Wang, Chenyi Liu, Minlie Huang, and Liqiang Nie. 2018. Learning to Ask Questions in Open-domain Conversational Systems with Typed Decoders. In *Proceedings of ACL*.
[14] Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8, 3-4 (1992), 229–256.
[15] Min Yang, Qiang Qu, Ying Shen, Qiao Liu, Wei Zhao, and Jia Zhu. 2018. Aspect and sentiment aware abstractive review summarization. In *Proceedings of COLING*.
[16] Min Yang, Qiang Qu, Jia Zhu, and et al. 2018. Cross-domain Aspect/Sentiment-aware Abstractive Review Summarization. In *Proceedings of CIKM*.