Paper Abstract Writing through Editing Mechanism

Qingyun Wang[1*], **Zhihao Zhou**[1*], **Lifu Huang**[1], **Spencer Whitehead**[1],
Boliang Zhang[1], **Heng Ji**[1], **Kevin Knight**[2]
[1] Rensselaer Polytechnic Institute
jih@rpi.edu
[2] University of Southern California
knight@isi.edu

Abstract

We present a paper abstract writing system based on an attentive neural sequence-to-sequence model that can take a title as input and automatically generate an abstract. We design a novel Writing-editing Network that can attend to both the title and the previously generated abstract drafts and then iteratively revise and polish the abstract. With two series of Turing tests, where the human judges are asked to distinguish the system-generated abstracts from human-written ones, our system passes Turing tests by junior domain experts at a rate up to 30% and by non-expert at a rate up to 80%.[1]

1 Introduction

Routine writing, such as writing scientific papers or patents, is a very common exercise. It can be traced back to the "*Eight legged essay*", an austere writing style in the Ming-Qing dynasty.[2] We explore automated routine writing, with paper abstract writing as a case study. Given a title, we aim to automatically generate a paper abstract. We hope our approach can serve as an assistive technology for human to write paper abstracts more efficiently and professionally, by generating an initial draft for humans further editing, correction and enrichment.

A scientific paper abstract should always **focus on the topics** specified in the title. However, a typical recurrent neural network (RNN) based approach easily loses focus. Given the title "*An effective method of using **Web** based **information** for **Relation Extraction***" from Keong and Su (2008), we compare the human written abstract and system generated abstracts in Table 1. The *LSTM_LM* baseline generated abstract misses the key term "*Web*" mentioned in the paper title. We introduce a title attention (Bahdanau et al., 2015; Luong et al., 2015) into a sequence-to-sequence model (Sutskever et al., 2014; Cho et al., 2014) to guide the generation process so the abstract is topically relevant to the given title, as shown in the "Seq2seq with attention" row of Table 1.

Previous work usually models natural language generation as a one-way decision problem, where models generate a sequence of tokens as output and then moves on, never coming back to modify or improve the output. However, human writers usually start with a draft and keep polishing and revising it. As C. J. Cherryh once said, "it is perfectly okay to write garbage - as long as you edit brilliantly." [3] We model abstract generation as a conditioned, iterative text generation problem and design a new **Writing-editing Network** with an **Attentive Revision Gate** to iteratively examine, improve, and edit the abstract with guidance from the paper title as well as the previously generated abstract. A result of the Writing-editing Network is shown in Table 1, where we can see that the initial draft contains more topically relevant and richer concepts than the title, such as the term '*IE*'. By adding this initial draft as feedback and guidance, it eases the next generation iteration, allowing the model to focus on a more limited learning space, and generate more concise and coherent abstracts.

[*] Qingyun Wang and Zhihao Zhou contributed equally to this work.

[1] The datasets and programs are publicly available for research purpose https://github.com/EagleW/Writing-editing-Network

[2] https://en.wikipedia.org/wiki/Eight-legged_essay

[3] https://www.goodreads.com/quotes/398754-it-is-perfectly-okay-to-write-garbage–as-long-as-you

Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Short Papers), pages 260–265
Melbourne, Australia, July 15 - 20, 2018.

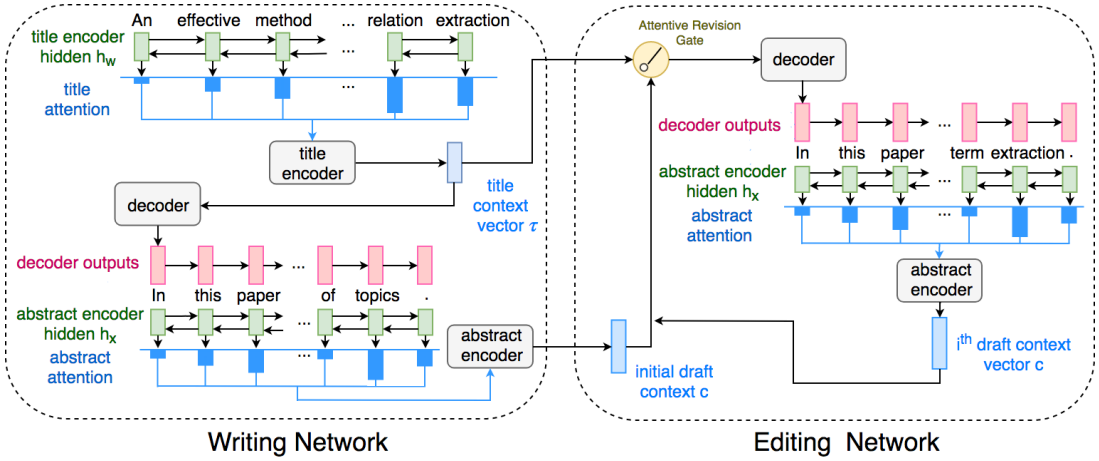

Figure 1: Writing-editing Network architecture overview.

Title	An effective method of using **Web** based **information** for **Relation Extraction** (Keong and Su, 2008)
Human written abstract	*We propose a method that incorporates paraphrase **information** from the **Web** to boost the performance of a supervised **relation extraction** system. Contextual **information** is extracted from the **Web** using a semi-supervised process, and summarized by skip-bigram overlap measures over the entire **extract**. This allows the capture of local contextual **information** as well as more distant associations. We observe a statistically significant boost in **relation extraction** performance.*
LSTM_LM	*This paper proposes a method for automatic **extraction** of salient **information** from an original text. Our method shows promising results in which a noun clustering is employed. The **relation** is then be applied to the **relation extraction** task.*
Seq2seq with attention (Initial Draft)	*In this paper we focus on the task of **extracting** the most commonly used **Web** pages in named entity recognition (IE) **information** from a large corpus of news articles. In particular, we propose a new kernel method that can be applied to a large collection of **Web** pages by automatic detection of labeled and unlabeled data. Our method is very simple and because the related method of using the data to be used as a guide as a source of **information** for the topic detection and classification can be applied, but often fail at improving the performance by combining **relations** and previous methods. In this method, the co-occurrence counts and the occurrences of each **relation** are not statistically significant, but also that it is possible to contribute to entity **relations** with a term in a variety of topics.*
Writing-editing Networks (Final Draft)	*In this paper we focus on the task of identifying the most commonly relevant features of **Web** documents. In particular, we propose a generic, automated IE algorithm that can be applied to a large collection of **Web** pages containing full large documents. This is a first step in helping a wide range of collaborative works for **relation extraction**. We show that it is possible to eliminate a good number of errors in **relation extraction** from a variety of documents, but that it is difficult to define a problem of term **extraction**.*

Table 1: Human and system generated abstracts for the same title.

2 Approach

In this section, we describe our "Writing-editing Network" (Figure 1). The writing network takes a title as input and generates the first abstract draft. The editing network takes both the title and previous draft as input to iteratively proof-read, improve, and generate new versions.

2.1 Writing Network

Our Writing Network is based on an attentive sequence-to-sequence model. We use a bi-directional gated recurrent unit (GRU) (Cho et al., 2014) as an encoder, which takes a title $\mathcal{T} = \{w_1, \ldots, w_K\}$ as input. For each token, w_k, the encoder produces a hidden state, h_{w_k}.

We employ a GRU as our decoder to generate the draft abstract $X^{(0)} = \{x_1^{(0)}, \ldots, x_N^{(0)}\}$. To capture the correlation between the title, $\mathcal{T}$, and the abstract draft, $X^{(0)}$, we adopt a soft-alignment attention mechanism (Bahdanau et al., 2015), which enables the decoder to focus on the most relevant words from the title. At the n^{th} decoder step, we apply the soft attention to the encoder hidden states to obtain an attentive title context vector, τ_n:

$$\tau_n = \sum_{k=1}^{K} \alpha_{n,k} h_{w_k}$$
$$\alpha_{n,k} = \text{softmax}\left(f\left(s_{n-1}, h_{w_k}\right)\right) \quad (1)$$

where s_{n-1} is the $n-1^{th}$ hidden state, $s_0 = h_{w_K}$ which is the last hidden state of the encoder, f is a function that measures the relatedness of word w_k in the title and word $x_{n-1}^{(0)}$ in the output abstract.

The decoder then generates the n^{th} hidden state, s_n, which is given by:

$$s_n = \text{GRU}(x^{(0)}_{n-1}, s_{n-1}, \tau_n)$$
$$p(x^{(0)}_n | x^{(0)}_{1:n-1}, w_{1:K}) = g(x^{(0)}_{n-1}, s_n, \tau_n) \quad (2)$$

where the function g is a softmax classifier, which is used to find the next word, $x^{(0)}_n$, by selecting the word of maximum probability.

2.2 Editing Network

The concepts contained in the titles are usually limited, so the learning space for the generator is huge, which hinders the quality of the generated abstract. Compared to the title, the generated abstracts contain more topically relevant concepts, and can provide better guidance. Therefore, we design an Editing Network, which, besides the title, also takes the previously generated abstract as input and iteratively refines the generated abstract. The Editing Network follows an architecture similar to the Writing Network.

Given an initial draft, $X^{(0)}$, from the Writing Network, we use a separate bi-directional GRU encoder, to encode each $x^{(0)}_n \in X^{(0)}$ into a new representation, $h_{x^{(0)}_n}$. As in the Writing Network, we use $s_0 = h_{w_K}$ as the initial decoder hidden state of the Editing Network decoder, which shares weights with the Writing Network decoder.

At the n^{th} decoder step, we compute an attentive draft context vector, c_t, by applying the same soft attention function from Eq. (1) to the encoded draft representations, $\{h_{x^{(0)}_1}, \ldots, h_{x^{(0)}_N}\}$, using decoder state s_{n-1}.[4] We also recompute the attentive title context vector, τ_n, with the same soft attention, though these attentions do not share weights. Intuitively, this attention mechanism allows the model to proofread the previously generated abstract and improve it by better capturing long-term dependency and relevance to the title. We incorporate c_t into the model through a novel **Attentive Revision Gate** that adaptively attends to the title and the previous draft at each generation step:

$$r_n = \sigma\left(W_{r,c}c_n + W_{r,\tau}\tau_n + b_r\right) \quad (3)$$
$$z_n = \sigma\left(W_{z,c}c_n + W_{z,\tau}\tau_n + b_z\right) \quad (4)$$
$$\rho_n = \tanh\left(W_{\rho,c}c_n + z_n \odot \left(W_{\rho,\tau}\tau_n + b_\rho\right)\right) \quad (5)$$
$$a_n = r_n \odot c_n + (1 - r_n) \odot \rho_n \quad (6)$$

[4]The indices are changed since the generated sequence lengths from the writing and editing networks may differ.

where all W and b are learned parameters. With the attention vector, a_n, we compute the n^{th} token with the same decoder as in section 2.1, yielding another draft $X^{(1)} = \{x^{(1)}_1, \ldots, x^{(1)}_T\}$. We repeat this process for d iterations. In our experiments, we set d to 2 and found it to work best.

3 Experiments

3.1 Data and Hyperparameters

We select NLP as our test domain because we have easy access to data and domain experts for human judges. We collected a data set of 10,874 paper title and abstract pairs[5] from the ACL Anthology Network[6] (until 2016) for our experiments. We randomly dividing them into training (80%), validation (10%), and testing (10%) sets. On average, each title and abstract include 9 and 116 words, respectively. Our model has 512 dimensional word embeddings, 512 encoder hidden units, and 1,024 decoder hidden units.

3.2 Method Comparison

Method	METEOR	ROUGE-L	HUMAN PREFERENCE
LSTM-LM	8.7	15.1	0
Seq2seq	13.5	19.2	22
ED(1)	13.3	**20.3**	30
ED(2)	**14.0**	19.8	**48**

Table 2: Method Comparison (%).

n	1	2	3	4	5	6
System	100	94.4	67.3	35.0	15.9	6.6
Human	98.2	78.5	42.2	17.9	7.7	4.1

Table 3: Plagiarism Check: Percentage (%) of n-grams in test abstracts generated by system/human which appeared in training data.

We include an LSTM Language Model (Sundermeyer et al., 2012) (*LSTM-LM*) and a Seq2seq with Attention (*Seq2seq*) model as our baselines and compare them with the first (*ED(1)*) and second revised draft (*ED(2)*) produced by the Writing-editing Network.

Table 2 presents METEOR (Denkowski and Lavie, 2014) and ROUGE-L (Lin, 2004) scores for each method, where we can see score gains on

[5]https://github.com/EagleW/ACL_titles_abstracts_dataset

[6]http://clair.eecs.umich.edu/aan/index.php

	# Tests	# Choices per Test	Non-expert		NLP Expert	
			Non-CS	CS	Junior	Senior
Different Titles	50	2	30%	15%	12%	0%
	20	5	60%	20%	30%	20%
	10	10	80%	30%	30%	20%
Same Title	50	2	54%	10%	4%	0%
	20	5	75%	25%	5%	5%

Table 4: Turing Test Passing Rates.

both metrics from the Editing Mechanism. Additionally, 10 NLP researchers manually assess the quality of each method. We randomly selected 50 titles and applied each model to generate an abstract. We then asked human judges to choose the best generated abstract for each title and computed the overall percentage of each model being preferred by human, which we record as *Human Preference*. The criteria the human judges adopt include topical relevance, logical coherence, and conciseness. Table 2 shows that the human judges strongly favor the abstracts from our ED(2) method.

We also conduct a plagiarism check in Table 3, which shows that 93.4% of 6-grams generated by ED(2) did not appear in the training data, indicating that our model is not simply copying. The 6-grams borrowed by both our model and human include "*word sense disambiguation (wsd)*", "*support vector machines (svm)*", "*show that our approach is feasible*", and "*we present a machine learning approach*". However, human writing is still more creative. The uni-grams and bi-grams that appear in human written test abstracts but not in the training set include "*android*", "*ascii*", '*p2p*", "*embellish*", "*supervision bottleneck*", "*medical image*", "*online behaviors*", and "*1,000 languages*".

3.3 Impact of Editing Mechanism

n	1	2	3	4	5	6
METEOR	13.3	**14.0**	13.6	13.9	13.8	13.5
ROUGE-L	**20.3**	19.8	18.6	19.2	18.9	18.8

Table 5: Iteration comparison (%)

We trained and evaluated our editing approach with 1-6 iterations and the experimental results (Table 5) showed that the second iteration produced the best results. The reason may be as follows. The attentive revision gate incorporates the knowledge from the paper title and the previous generated abstract. As the editing process iterates, the knowledge pool will diverge since in each iteration the generated abstract may introduce some irrelevant information. Empirically the second iteration achieved a good trade-off between good quality of generated abstract and relevance with topics in the title.

3.4 Turing Test

We carried out two series of Turing tests, where the human judges were asked to distinguish the fake (system-generated) abstracts from the real (human-written) ones. (1)**Abstracts for different titles**. We asked the human judges to identify the fake abstract from a set of $N-1$ real ones (i.e., N choose 1 question). A test is passed when a human judge mistakenly chooses a real abstract. (2) **Abstracts for the same title**. We asked the human judges to choose the real abstract from a set of $N-1$ fake ones. A test is passed when a human judge mistakenly chooses a fake abstract.

As expected, Table 4 shows that people with less domain knowledge are more easily deceived. Specifically, non-CS human judges fail at more than half of the 1-to-1 sets for the same titles, which suggests that most of our system generated abstracts follow correct grammar and consistent writing style. Domain experts fail on 1 or 2 sets, mostly because the human written abstracts in those sets don't seem very topically relevant. Additionally, the more abstracts that we provided to human judges, the easier it is to conceal the system generated abstract amongst human generated ones.

A human is still more intelligent than the machine on this task from many reasons: (1) Machines lack knowledge of the deep connections among scientific knowledge elements and thus produce some fluent but scientifically incorrect concepts like "*...a translation system to generate a parallel corpus...*" and "*...automatic generation of English verbs...*". (2) Humans know better about what terms are more important than others in a title. For example, if a language name ap-

pears in the title, it must appear in the abstract. We have an automatic term labeling approach, but, unfortunately, its performance (75% F-score) is not good enough to help the abstract generation. (3) Human written abstracts are generally more specific, concise, and engaging, often containing specific lab names, author names (e.g., "*Collins proposed...*"), system abbreviations, and terminologies (e.g., "*Italian complex nominals (cns) of the type n+p+n*"). In contrast, our system occasionally generates too general descriptions like "*Topic modeling is a research topic in Natural Language Processing.*" (4) Machines lack common sense knowledge, so a system generated abstract may mention three areas/steps, but only outline two of them. (5) Machines lack logical coherence. A system generated abstract may contain "*The two languages...*" and not state which languages. (6) We are not asking the system to perform scientific experiments, and thus the system generated "experimental results" are often invalid, such as "*Our system ranked first out of the participating teams in the field of providing such a distribution.*".

4 Related work

Deep neural networks are widely applied to text generation tasks such as poetry creation (Greene et al., 2010; Ghazvininejad et al., 2016; Zhang et al., 2017), recipe generation (Kiddon et al., 2016), abstractive summarization (Gu et al., 2016; Wang and Ling, 2016; See et al., 2017), and biography generation (Lebret et al., 2016; Liu et al., 2018). We introduce a new task of generating paper abstracts from the given titles. We design a Writing-editing Network which shares ideas with Curriculum Learning (Bengio et al., 2009), where training on a data point from coarse to fine-grained can lead to better convergence (Krueger and Dayan, 2009). Our model is different from previous theme-rewriting (Polozov et al., 2015; Koncel-Kedziorski et al., 2016) approach which has been applied to math word problems but more similar to the Feedback Network (Zamir et al., 2017) by using previous generated outputs as feedback to guide subsequent generation. Moreover, our Writing-editing Network treats previous drafts as independent observations and does not propagate errors to previous draft generation stages. This property is vital for training feedback architectures for discrete data. Another similar approach is the deliberation network used for Machine Translation (Xia et al., 2017). Instead of directly concatenating the output of the encoder and writing network, we use the learnable Attentive Revision Gate to control their integration.

5 Conclusions and Future Work

We propose a new paper abstract generation task, present a novel Writing-editing Network architecture based on an Editing Mechanism, and demonstrate its effectiveness through both automatic and human evaluations. In the future we plan to extend the scope to generate a full paper by taking additional knowledge bases as input.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*.

Marjan Ghazvininejad, Xing Shi, Yejin Choi, and Kevin Knight. 2016. Generating topical poetry. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

Erica Greene, Tugba Bodrumlu, and Kevin Knight. 2010. Automatic analysis of rhythmic poetry with applications to generation and translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*.

Stanley Yong Wai Keong and Jian Su. 2008. An effective method of using web based information for relation extraction. In *Proceedings of the Third International Joint Conference on Natural Language Processing*.

Chloé Kiddon, Luke Zettlemoyer, and Yejin Choi. 2016. Globally coherent text generation with neural checklist models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

Rik Koncel-Kedziorski, Ioannis Konstas, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2016. A theme-rewriting approach for generating algebra word problems. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

Kai A Krueger and Peter Dayan. 2009. Flexible shaping: How learning in small steps helps. *Cognition*, 110(3):380–394.

Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Proceedings of Text Summarization Branches Out*.

Tianyu Liu, Kexiang Wang, Lei Sha, Baobao Chang, and Zhifang Sui. 2018. Table-to-text generation by structure-aware seq2seq learning. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.

Oleksandr Polozov, Eleanor O'Rourke, Adam M. Smith, Luke Zettlemoyer, Sumit Gulwani, and Zoran Popovic. 2015. Personalized mathematical word problem generation. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*.

Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. 2012. Lstm neural networks for language modeling. In *Proceedings of Thirteenth Annual Conference of the International Speech Communication Association*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing systems*.

Lu Wang and Wang Ling. 2016. Neural network-based abstract generation for opinions and arguments. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Yingce Xia, Fei Tian, Lijun Wu, Jianxin Lin, Tao Qin, Nenghai Yu, and Tie-Yan Liu. 2017. Deliberation networks: Sequence generation beyond one-pass decoding. In *Advances in Neural Information Processing Systems 30*.

Amir R. Zamir, Te-Lin Wu, Lin Sun, William B. Shen, Bertram E. Shi, Jitendra Malik, and Silvio Savarese. 2017. Feedback networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Jiyuan Zhang, Yang Feng, Dong Wang, Yang Wang, Andrew Abel, Shiyue Zhang, and Andi Zhang. 2017. Flexible and creative chinese poetry generation using neural memory. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*.