Keeping Consistency of Sentence Generation and Document Classification with Multi-Task Learning

Toru Nishino **Shotaro Misawa** **Ryuji Kano**
Tomoki Taniguchi **Yasuhide Miura** **Tomoko Ohkuma**
Fuji Xerox Co., Ltd.
{nishino.toru, misawa.shotaro, kano.ryuji,
taniguchi.tomoki, yasuhide.miura, ohkuma.tomoko}
@fujixerox.co.jp

Abstract

The automated generation of information indicating the characteristics of articles such as headlines, key phrases, summaries and categories helps writers to alleviate their workload. Previous research has tackled these tasks using neural abstractive summarization and classification methods. However, the outputs may be inconsistent if they are generated individually. The purpose of our study is to generate multiple outputs consistently. We introduce a multi-task learning model with a shared encoder and multiple decoders for each task. We propose a novel loss function called *hierarchical consistency loss* to maintain consistency among the attention weights of the decoders. To evaluate the consistency, we employ a human evaluation. The results show that our model generates more consistent headlines, key phrases and categories. In addition, our model outperforms the baseline model on the ROUGE scores, and generates more adequate and fluent headlines.

1 Introduction

Headlines and other information such as key phrases, summaries and categories about articles are crucial for readers to search articles on demand. To attract more readers, writers manually create headlines and summaries by summarizing the articles, extract key phrases and classify articles into categories. Figure 1 shows an example of job advertisement articles. In addition to the job description text, the headline, key phrase and category are labeled for each article. Thus job seekers can easily retrieve the job advertisements they desire by reading them. However, it is a burden for writers to create these headlines, key phrases and categories manually for extremely large numbers of articles. Hence, an automatic generation system is highly demanded.

Headline: We Want Android[1] Engineer to Develop our New Service "Wantedly People[2] "!
Key Phrase: Android Engineer
Category: Engineer
Description Text (Truncated): We released a new service "Wantedly People" in November 2016, and released new features in July 2017! Our "People team", developing rapidly growing "Wantedly People" service, is recruiting an Android Engineer! If you love to work on new services, or if you want to try to develop new apps as your representative work, we are looking forward to your application!

Figure 1: An example job advertisement article with consistency. Each article contains a headline, key phrase and category. The underlined topic "Engineer" is consistently noted in the headline, key phrase and category. A Japanese-English translation is applied.

For automated generation of multiple outputs, consistency among outputs is crucial. A lack of consistency among outputs causes incorrect information in the outputs. In Figure 1, for example, the "Engineer" position is consistently noted in the headline, key phrase and category. An occupation is consistent and salient information for headlines, key phrases and categories. If the article was to be misclassified as a "Designer" category or the key phrase wrongly noted as "Robotics Engineer," an inconsistency among the headline, key phrase and category would occur. Thus, readers would be confused by these inconsistencies. We must force generators to predict multiple outputs consistently. This leads to the correctness of the occupation in the outputs, and thus the quality of the generated outputs also improves.

In previous research, neural networks have achieved significant improvements in individual tasks, such as abstractive summarization (Rush et al., 2015; See et al., 2017; Shi et al., 2018),

[1]Android is a trademark of Google LLC.
[2]Wantedly People is a web service provided by Wantedly, Inc.

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pages 3195–3205,
Hong Kong, China, November 3–7, 2019.

headline generation (Takase et al., 2016), key phrase generation (Meng et al., 2017) and text classification (Zhang et al., 2015) tasks. However, consistency among multiple outputs is not considered in the strategy to predict multiple outputs with separate models.

The purpose of our study is to maintain consistency among automatic generated sentences and classified categories. We adopt multi-task learning (Caruana, 1997) to predict the headlines, key phrases and categories of articles in one unified model. We handle the key phrase generation and classification tasks not as auxiliary tasks, but as the desired outputs of our system. Multi-task learning enables the encoder to focus on the common and salient features in the input text.

We propose a novel *hierarchical consistency loss* to maintain consistency among the multiple outputs. A hierarchical consistency loss forces the attention weights of the decoders to focus on the same words in the input text, considering the hierarchical relation among tasks. There is a hierarchical relation among the tasks; the headline generator generally focuses on the wider range of words including the key phrase generator focus upon. In addition, this loss has a flexibility that alleviate the influences of errors propagated from other tasks, similar to soft-parameter sharing methods (Guo et al., 2018).

We design human evaluations using crowdsourcing to score the following three metrics: fluency, adequacy and consistency among the outputs. We implement human evaluations of the job advertisement dataset, and the results indicate that our model improves not only the consistency score but also the fluency and adequacy scores.

In addition, we conduct automatic evaluations of the job advertisement dataset and the modified CNN-DailyMail (CNN-DM) dataset (Nallapati et al., 2016). The automatic evaluations show that our method improves the ROUGE metric scores on both datasets, which has multiple outputs.

Overall, our contributions are as follows:

- We propose a multi-task sentence generation and document classification model.
- A novel *hierarchical consistency loss* is introduced to train the weights of attention to focus more on the same part of the input text among the task-specific decoders.

Our designed human evaluations show that our model generates more consistent outputs. Our proposed model generates more adequate and fluent outputs on a human evaluation, and achieves the best ROUGE score on an automatic evaluation.

2 Related Work

Abstractive summarization. Abstractive summarization is a task to generate a short summary that captures the core meaning of the original text. Rush et al. (2015) used a neural attention model, and See et al. (2017) introduced a pointer-generator network to copy out-of-vocabulary (OOV) words from the input text. Hsu et al. (2018) combined abstractive and extractive summarization with an inconsistency loss to encourage consistency between word-level attention weights of the abstracter and sentence-level attention weights of the extractor. Abstractive summarization techniques are generally applied to a headline generation because this is a similar task (Shen et al., 2017; Tan et al., 2017).

Multi-task learning. Multi-task learning, which trains different tasks in one unified model, has achieved success in many natural language processing tasks (Luong et al., 2016; Hashimoto et al., 2017; Liu et al., 2019). Typical multi-task learning models have a structure with a shared encoder to encode the input text and multiple decoders to generate outputs of each task. Multi-task learning has a benefit in that the shared encoder captures common features among tasks; in addition, the encoder focuses more on relevant and beneficial features, and disregards irrelevant and noisy features (Ruder, 2017).

Although a multi-task learning model is beneficial in training a shared encoder, it is still difficult to share information among task-specific decoders. Some studies have constructed a multi-task learning model using techniques that encourages information sharing among decoders. Isonuma et al. (2017) proposed an extractive summarization model that the outputs of the sentence extractor are directly used for a document classifier. Anastasopoulos and Chiang (2018) introduced a triangle model to transfer the decoder information of the second task to the decoder of the first task. Tan et al. (2017) introduced a coarse-to-fine model to generate headlines using important sentences chosen in the extracter. These methods are cascade models that additionally input the information of the first tasks directly into

the second tasks. They consider the hierarchy among tasks, but these models suffer from the errors of the previous tasks.

Guo et al. (2018) proposed a decoder sharing method with soft-parameter sharing to train the summarization and entailment tasks. Soft-parameter sharing has a benefit in that it provides more flexibility between the layer of summarization and entailment tasks; however, this method does not consider the hierarchy among tasks.

Our study extends the method in Hsu et al. (2018) to a multi-task learning model in which the models need to generate multiple outputs with consistency. Hierarchical consistency loss combines two advantages. This loss considers the hierarchy among tasks, and has flexibility among tasks, similar to soft-parameter sharing methods. We assess the advantages of this loss in Section 4.2.

3 Method

3.1 Problem Definition

We define the tasks of our study and describe the overview of the datasets.

Let $x = \{x_1, x_2, ...x_S\}$ be a sequence of input text. The target of our multi-task model is to generate two types of sentences and to predict the category of the input article. Our model predicts the exactly one category tag y^1 for each input sentence. Our model also predicts $y^2 = \{y^2_1, y^2_2, ...y^2_{T^2}\}$ and $y^3 = \{y^3_1, y^3_2, ...y^3_{T^3}\}$, which are the sequence of sentences.

For the job advertisement dataset, the targets of our model are to classify articles into occupation categories y^1 (task 1), generate key phrases regarding the occupation y^2 (task 2) and generate headlines y^3 (task 3). For the CNN-DM dataset, the targets are to predict the article categories y^1 (task 1), headlines y^2 (task 2) and multi-sentence summaries y^3 (task 3).

Here, S is the length of the input texts, and T^2 and T^3 are the lengths of the output sequences, respectively. T^2 is generally smaller than T^3 in both datasets. Hence, task 3 is generally more difficult than task 2, and more information is needed to generate y^3 than y^2.

3.2 Encoder-Decoder model

Encoder-decoder model with attention mechanism. Our model is based on an encoder-decoder model (Cho et al., 2014). The encoder RNN transforms the input text into hidden vectors $h^e = \{h^e_1, h^e_2, ..., h^e_S\}$, and the decoder RNN then predicts the generation probability of each word $P_{vocab,t}$:

$$h^e_t = \mathrm{RNN}_{enc}(x_t, h^e_{t-1}) \quad (1)$$
$$h^d_t = \mathrm{RNN}_{dec}(y_{t-1}, h^d_{t-1}, h^e_S) \quad (2)$$
$$P_{vocab,t} = \mathrm{softmax}(W_{d2v} h^d_t + b_{d2v}) \quad (3)$$

where the weight matrices W_{d2v} and bias vector b_{d2v} are trainable parameters.

Rush et al. (2015) used an attention mechanism to handle long input sentences. The attention mechanism obtains the hidden vector of attention $\bar{h}^d_t$ from the hidden vector and context vector c^e_t, which is defined as the weighted sum of the hidden vectors of the encoder:

$$e^e_{tj} = v^T \tanh(W_e h^e_j + W_d h^d_t + b_{attn}) \quad (4)$$
$$\alpha^e_{tj} = \mathrm{softmax}(e^e_{tj}) \quad (5)$$
$$c^e_t = \sum_j \alpha^e_{tj} h^e_j \quad (6)$$
$$\bar{h}^d_t = W_c[h^d_t, c^e_t] + b_c \quad (7)$$

Note that $[h^d_t, c^e_t]$ indicates the concatenation of vectors h^d_t and c^e_t. Weight matrices W_e, W_d and W_c and the bias vectors b_{attn} and b_c are trainable parameters.

Pointer-generator network. We adopt a pointer-generator network (See et al., 2017) for the decoders to handle OOV words. The decoder generates words under the probability of p_{gen,y_t} and copies words from the input sentence under the probability of $1 - p_{gen,y_t}$.

Coverage mechanism. See et al. (2017) also introduced a coverage mechanism to alleviate the repetition problem. The coverage loss L_{cov} is added to the loss function to penalize the attention mechanism to avoid focusing on the same input words.

3.3 Multi-Task Learning for Generation and Classification

We introduce multi-task learning to predict multiple outputs simultaneously in one unified model. Figure 2 describes an overview of our multi-task learning model. A multi-task learning model comprises one shared layer, two task-specific decoders for generation tasks, and one classifier.

Shared encoder. First, shared encoder RNN_{enc} transforms the input text into the shared hidden

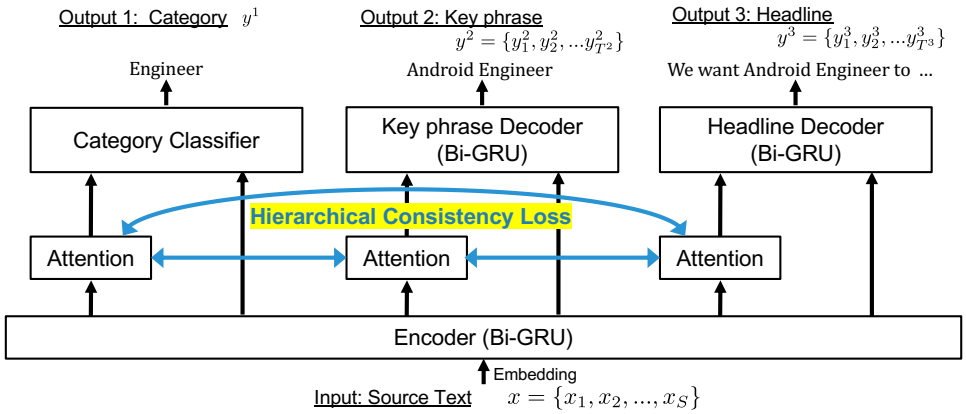

Figure 2: An overview of our multi-task learning model. The shared embedding layer and encoder first transform the input text into a shared hidden vector, and the task-specific layers then predict the outputs of each task. Hierarchical consistency loss penalizes the inconsistency between the attention weights of the pairs of decoders.

vector h_t^e. We use a 2-layer bi-directional GRU (Cho et al., 2014) for the encoder.

Classifier. The classifier transforms a shared hidden vector into the category probability P_{cat}. We implement the classifier as 2-layer perceptron with an attention mechanism:

$$o^c = \mathrm{ReLU}(W_1^c([h_S^e, c_t^e] + b_1^c)) \quad (8)$$

$$P_{cat} = \mathrm{softmax}(W_2^c o^c + b_2^c) \quad (9)$$

where the weight matrices W_1^c and W_2^c, and bias vectors b_1^c and b_2^c are trainable parameters.

Decoders. The hidden vector of encoder h_S^e is first transformed into $\hat{h_0^d}$ in the bridge layer, and the decoders RNN_{dec} of each task then generate output sequences y^2 and y^3 from $\hat{h_0^d}$. A bridge is an additional fully connected layer used to fit the hidden vector into each task. We expect that multi-task learning will enable the shared encoder to capture more common and salient parts of an article, that is, the parts of the text that mention the occupation.

3.4 Hierarchical Consistency Loss

The main objective of our method is to maintain consistency among multiple outputs. If each decoder focuses on the same word in the input text, the model can generate a more consistent output. Hence, consistency between attention weights leads to consistency between multiple outputs. For example, in Figure 3, inconsistency regarding the occupation occurs because the word "engineer" is inconsistently focused on the attention weights of the key phrase generator. From this, the key phrase generator predicts an incorrect key phrase. By penalizing such inconsistencies, the model enables the generation of more consistent outputs.

However, perfect consistency between attention weights occasionally disturbs the model to generate proper outputs. Because headlines contain more information than key phrases, the headline generator must focus on a wider range of words in the input text than the key phrase generator. For example, in Figure 3, the headline generator focuses on the words "user interface." In contrast, the key phrase generator does not focus on these words. Therefore, the attention weights among tasks generally maintain a hierarchical relation. That is, the higher tasks (headline generation and summarization) need to focus on a wider range of words, and the lower tasks (classification) focus on the range on which the higher tasks focused.

We introduce a novel "hierarchical consistency loss" to penalize inconsistency among multiple outputs. We define hierarchical consistency loss between task s and task t as follows:

$$L_{hcl}^{st} = \frac{\lambda_{hcl}}{S} \sum_{i=1}^{S} |\max_j e_{ij}^{ds} - \max_j e_{ij}^{dt}|_+ \quad (10)$$

where e_{ij}^{ds} is the non-normalized attention weight of the output word j toward the input word i. Note that a ramp function $|x|_+$ is used to compare two attention weights. Task t is a task that needed to focus on a wider range of words than the input task s. For example, in Figure 3, task s is a key phrase generation, and task t is a headline generation.

The aim of hierarchical consistency loss is to penalize inconsistency between two attention weights. For example, in Figure 3, the loss forces the attention weight corresponding to the word

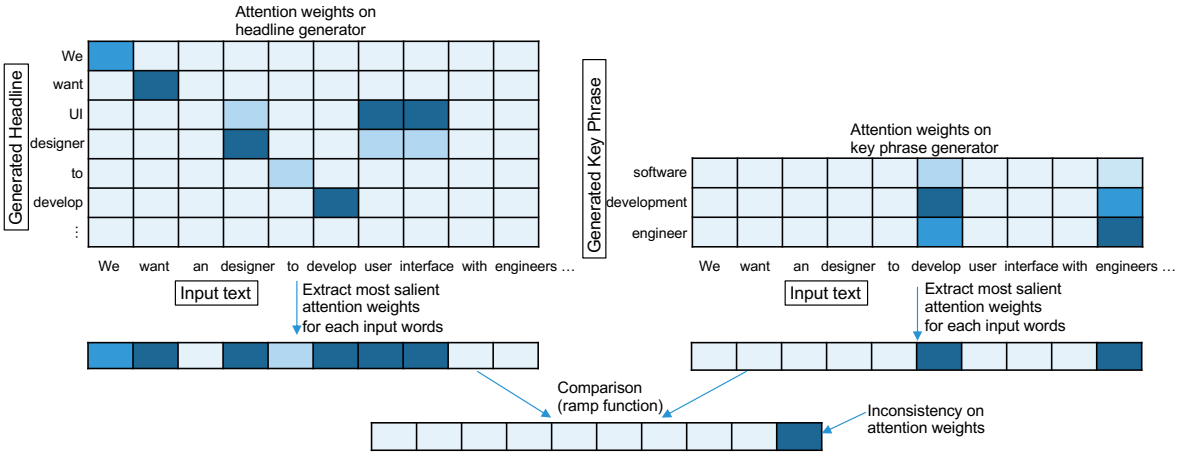

Figure 3: An overview illustration of our hierarchical consistency loss. This loss penalizes a case in which inconsistency occurs between two outputs. The decoder of the key phrase generator focuses on the word "engineers." However, the headline generation decoder does not focus on this word. Hierarchical consistency loss treats this difference as an inconsistency.

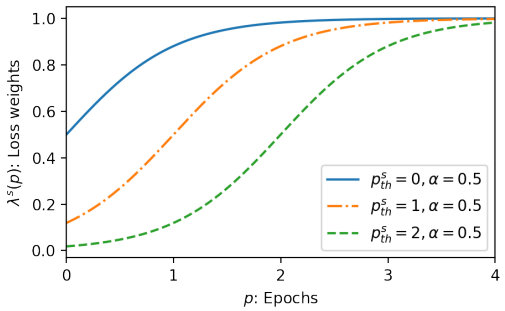

Figure 4: An illustration of our training scheduling strategy. The loss weights $\lambda^s(p)$ gradually increase as the training proceeds.

"engineers" to a higher weight because the word "engineers" is crucial to generate consistent and adequate outputs. Figure 3 shows the process of the loss. The most salient attention weights for each input word are extracted, and the extracted attention weights between two tasks are then compared. A ramp function enables this loss to consider the hierarchical relation between two tasks.

3.5 Learning Scheduling for Multi-Task Learning

We also introduce a new multi-task learning scheduling strategy to effectively train several tasks with different levels of difficulty. Kiperwasser and Ballesteros (2018) introduced a strategy that gradually changes the probability that the training examples will be picked from each dataset as the training proceeds. We modify this idea to a situation in which the input text is in common among multiple tasks, adjusting the hyperparameters of the weighted sum of loss functions. This strategy enables us to train our model in such a way that our model focuses on learning easy tasks at the beginning, and then gradually focuses more on learning difficult tasks.

We determine the weights of the loss function $\lambda^s(p)$ for task s with a sigmoid function:

$$\lambda^s(p) = \lambda^s_{const} \frac{1}{1 + \exp((p^s_{th} - p)/\alpha)} \quad (11)$$

where λ^s_{const} and p^s_{th} are hyperparameters for each task. Parameter p describes the number of epochs trained thus far. We set hyperparameter α to 0.5 for all tasks. As illustrated in Figure 4, $\lambda^s(p)$ becomes larger as the training proceeds.

3.6 Overall Loss Function

The overall loss function of the proposed model is calculated as the weight sum of losses from three tasks, coverage losses and hierarchical consistency loss:

$$\begin{aligned} L_{all} &= \lambda^1(p)L_1 + \lambda^2(p)L_2 + \lambda^3(p)L_3 \\ &+ \lambda^2_{cov}L^2_{cov} + \lambda^3_{cov}L^3_{cov} \\ &+ \lambda^{all}_{hcl}L^{all}_{hcl} \end{aligned} \quad (12)$$

where $\lambda^1(p)$, $\lambda^2(p)$, $\lambda^3(p)$, λ^2_{cov}, λ^3_{cov} and λ^{all}_{hcl} are hyperparameters. In addition, $\lambda^1(p)$, $\lambda^2(p)$, and $\lambda^3(p)$ are calculated using Eqn.11. Moreover, L_1, L_2 and L_3 signify the loss of the classification, key phrase generation and headline generation tasks, respectively. L^2_{cov} and L^3_{cov} are the coverage losses

of two generators, and L_{hcl}^{all} is the sum of the hierarchical consistency losses:

$$L_{hcl}^{all} = L_{hcl}^{12} + L_{hcl}^{13} + L_{hcl}^{23} \quad (13)$$

where L_{hcl}^{st} indicates the hierarchical consistency loss between task s and t.

4 Experiments and Results

4.1 Experimental Settings

We use two datasets, namely, a job advertisement dataset and the CNN-DailyMail (CNN-DM) dataset. We evaluate the headline generation, key phrase generation and classification tasks in these datasets. As an example, Figure 1 shows the headlines, key phrases and categories, which generally include common features, that is, the words mentioning occupations.

We also use the CNN-DM dataset (See et al., 2017) in an automatic evaluation under different task settings. We evaluate the multi-sentence summarization, headline generation and classification tasks in this dataset. Note that we train and evaluate the CNN and DailyMail datasets separately because they have different taxonomies.

The original CNN-DM dataset does not contains headlines or categories, and thus we extract the headlines and categories from the raw HTMLs. The Supplementary section describes the details and modification of the CNN-DM dataset.

We employ pointer-generator models without multi-task learning as baselines of the sentence generators, and a 2-layer GRU with attention as the baseline of the classifier. We chose all hyperparameters based on the learning curve and the scores of the validation set. Details of our model, such as the hyperparameters of the RNNs, the weights of the loss function and the optimizers, are described in the Supplementary section for reproducibility.

4.2 Results

Human evaluation of job advertisement dataset. We conduct a human evaluation of the job advertisement dataset to measure the quality of the generated outputs. Ten crowd-sourcing workers measure 250 randomly selected samples. We defined the following four metrics:

- Consistency: Measure whether the outputs include the same occupation. We choose a pair of outputs, and if both outputs mention the same type of occupation, we regard this pair as consistent.
- Adequacy: Measure how well the headline describes the correct information.
- Fluency: Measure how natural the generated headline is.
- Occupation Adequacy: Measure how well the headline mentions the correct occupation. For example, in Figure 1, if the generated headline implies that the occupation "engineer" is recruited, we regard this headline as adequate regarding the occupation. We assume that consistency of the occupation improves the score of the occupation adequacy.

	HG-KG	HG-CC	KG-CC	3 Outputs
Baseline	56.8%	37.6%	37.6%	30.0%
Proposed	**58.8%**	**39.6%**	**39.2%**	**32.4%**
Gold	65.2%	44.4%	48.8%	35.2%

Table 1: Comparison of human evaluation results with their consistency. HG-KG, HG-CC and KG-CC indicate the consistency scores between the headline generation and key phrase generation, headline generation and category classification and key phrase generation and category classification, respectively.

	Adequacy	Fluency	Occupation Adequacy
Baseline	3.34	3.69	3.45
Proposed	**3.76**	**3.86**	**3.89**
Gold	4.09	4.12	4.13

Table 2: Comparison of human evaluation results for headlines. Scores are an average of ten crowd-sourcing workers with five scale rating.

First, we conduct a human evaluation to measure the consistency among three outputs. Table 1 shows the evaluation results with their consistency. The scores are the percentage of articles evaluated as consistent by a majority of workers. We consider all three outputs as consistent if all pairs of outputs are consistent. The results indicate that the proposed method improves the consistency of the three generated outputs [1].

Second, to evaluate the quality of the generated headline, we implement a human evaluation to measure the adequacy and fluency. Table 2 shows the evaluation result along with the adequacy and fluency. The proposed method improves the

[1]Note that the consistency of HG-CC and KG-CC is relatively low because some categories are too abstractive (for example, "other types of engineers").

	Headline generation			Key phrase generation			Classification
	R-1	R-2	R-L	R-1	R-2	R-L	Accuracy
Baseline (Pointer-Generator Network)	25.1	5.3	21.1	30.9	10.6	28.7	62.8
Multi-Task Learning (MTL)	26.2	5.8	21.6	32.3	10.9	30.0	64.1
MTL + Scheduling (SD)	26.3	6.0	21.8	32.3	10.4	29.9	63.9
Proposed (MTL + SD + Hierarchical Consistency Loss (HCL))	$*$**26.9**	$*$**6.1**	$*$**22.4**	$*$**32.8**	$*$**11.2**	$*$**30.5**	**64.4**
Lead-1	19.0	4.3	13.5	-	-	-	-

Table 3: Automatic evaluation results based on the ROUGE metrics and accuracy (%) of classification of job advertisement dataset. R-1, R-2 and R-L indicate the F_1 scores of ROUGE-1, ROUGE-2 and ROUGE-L, respectively. The proposed method (MTL + SD + HCL) achieved the best scores (bold) for all tasks, and the ROUGE scores are statistically significant from the baseline model ($p < 0.05$) in both the headline and key phrase (indicated as $*$). Lead-1 is the baseline score, which uses the first sentence of the input article as a predicted headline.

		Summarization			Headline Generation			Classification
		R-1	R-2	R-L	R-1	R-2	R-L	Accuracy
CNN	Baseline	30.7	10.6	27.3	19.5	5.0	17.0	43.8
	Proposed (MTL + SD + HCL)	$*$**31.0**	$*$**10.9**	$*$ **27.8**	**19.6**	**5.0**	**17.1**	**43.9**
	Lead	33.4	12.2	26.1	17.2	5.0	11.1	-
DailyMail	Baseline	38.4	15.8	35.0	43.1	25.3	39.6	89.0
	Proposed (MTL + SD + HCL)	$*$**38.9**	$*$**16.3**	$*$**35.4**	$*$**43.7**	**25.5**	$*$**40.1**	**89.8**
	Lead	43.8	19.2	37.3	27.7	10.9	21.7	-

Table 4: Automatic evaluation results based on the ROUGE metrics and accuracy (%) of classification of the CNN and DailyMail datasets. The metrics are the same as in Table 3. The proposed method (MTL + SD + HCL) improved the scores for all tasks. The scores with $*$ indicate that the scores are statistically significant from the baseline model ($p < 0.05$). The lead is the score that uses the first three sentences of the input article as a predicted summary, and the first sentence of the input article as a predicted headline.

adequacy by $0.42pt$ and the occupation adequacy by $0.44pt$. Proposed method can generate more adequate outputs, particularly for the occupation.

Automatic evaluation of job advertisement corpus. We implement an automatic evaluation using the ROUGE metrics (Lin, 2004) and accuracy. We conduct the experiment ten times, and calculate the average score. Table 3 shows the effect of the proposed methods: multi-task learning (MTL), scheduling strategy (SD) and hierarchical consistency loss (HCL). From this result, the proposed method (MTL + SD + HCL) achieves the best score on all three tasks. MTL and HCL improve for all three tasks, and SD improves the score of the headline generation.

Automatic evaluation of the CNN-DM dataset. Table 4 shows the results of the CNN and DailyMail datasets, respectively. For both datasets, the proposed method improves the ROUGE scores of the summarization and headline generation.

From Table 5, headlines and key phrases of the job advertisement dataset have more overlap than the summaries and headlines of the CNN-DM dataset. Therefore, tasks of the job advertisement dataset benefit more from maintaining the consistency. For this reason, the scores of the job advertisement dataset achieve greater improvement than the CNN-DM dataset.

	ROUGE-1	ROUGE-2	ROUGE-L
Job Ads	48.4	15.9	47.9
CNN	42.8	13.7	38.8
DailyMail	37.9	12.6	34.0

Table 5: ROUGE recall scores of task 2 (key phrases for the Job Ads, headlines for CNN-DM) gold sentences against task 3 (headlines for Job Ads, summaries for CNN-DM) gold sentences in the validation set. Higher scores indicate that words appearing in the sentence of task 2 are overlapped by the sentence of task 3.

Comparison of the decoder information sharing methods. To validate the advantages of our hierarchical consistency loss, we compare five decoder information sharing methods: a cascade model, soft-parameter sharing, non-hierarchical consistency loss, hierarchical consistency loss with normalized attention weights and our hierarchical consistency loss. Table 6 presents the comparison of the five methods for the job advertisement dataset.

The cascade model, similar to Isonuma et al.

	Headline Generation			Key Phrase Generation			Classification
	R-1	R-2	R-L	R-1	R-2	R-L	Accuracy
Baseline (Pointer-Generator Network)	25.1	5.3	21.1	30.9	10.6	28.7	62.8
Proposed (MTL + SD + HCL)	**26.9**	**6.1**	**22.4**	**32.8**	**11.2**	**30.5**	64.4
Comparison of Decoder Information Sharing Method							
MTL + SD	26.3	6.0	21.8	32.3	10.4	29.9	63.9
MTL + SD + Cascade Model	26.3	5.6	21.6	31.8	10.6	29.5	64.4
MTL + SD + Cascade Model (Gold)	26.5	5.8	21.9	32.8	10.4	30.3	**64.5**
MTL + SD + Soft-Parameter Sharing	25.8	5.9	21.4	32.1	10.0	29.6	64.0
MTL + SD + Non-Hierarchical Consistency Loss	25.9	6.0	21.4	32.6	10.9	30.2	64.0
MTL + SD + HCL with Normalized Attention Weights	26.2	6.0	21.7	31.9	10.5	29.5	63.9
Comparison of Encoder Information Sharing Method							
HCL (SD and MTL are not applied)	25.8	5.6	21.2	31.0	10.1	28.7	63.1
SD + HCL (MTL is not applied)	25.6	5.6	21.5	31.2	10.2	28.9	62.6

Table 6: Comparison of the decoder information sharing methods and encoder sharing methods for the job advertisement dataset. The metrics are the same as in Table 3. The proposed method (adopting HCL) achieved the best scores (bold) compared to the other sharing methods.

(2017); Anastasopoulos and Chiang (2018), uses the output of the classifier as an additional input of the headline and key phrase generators. Meanwhile, the cascade model (gold) uses the gold classification category as the additional input of the generators during training and inference.

It can be observed from Table 6 that the cascade model achieves a lower score than MTL + SD. However, the cascade model (gold) improves the ROUGE-L score. This result indicates that the classification error propagates to the generators when the cascade model is applied. Our proposed HCL has an advantage in that it does not suffer from an error of the classifier, and thus this method achieves the best score.

Furthermore, we compare our proposed HCL model with other settings. A soft-parameter sharing method (Guo et al., 2018) penalizes the difference between parameters in pairs of decoders. The non-hierarchical consistency loss is almost the same as the hierarchical consistency loss (Eqn. 10). We replace a ramp function in Eqn. 10 with an absolute value function. Neither the soft-parameter sharing method nor the non-hierarchical consistency loss has the ability to consider the hierarchy among tasks.

It can be observed from Table 6 that both methods achieved a lower score than the hierarchical consistency loss. This is because our hierarchical consistency loss enables the model to penalize a multi-task model with hierarchy among the tasks.

Although the HCL with normalized-attention weights adopts the hierarchical consistency loss model indicated in Eqn. 10, however, we substitute the normalized attention weights $\alpha_{ij}^{d^s}$ for non-normalized attention weights $e_{ij}^{d^s}$.

HCL with normalized attention weights is not as effective as HCL with non-normalized attention weights. Our HCL is based on the assumption that if one specific input word is important for both the shorter and longer text generation tasks, the attention weights of the words for the shorter text generation task would be smaller than the attention weights for the longer text generation task. However, the distributions of the normalized attention weights converge to a few words for the shorter text generation task; thus the normalized attention weights does not satisfy the assumption. As a result, our proposed HCL with non-normalized attention weights can accurately compute this inconsistency, contrary to the HCL with normalized attention weights.

Comparison of encoder information sharing methods. To determine the dependence of HCL on MTL, we conduct two experiments. HCL applies the hierarchical consistency loss without MTL and SD, while SD + HCL applies the scheduling sampling and hierarchical consistency loss; however, multi-task learning is not applied. Table 6 presents the results of two methods for the job advertisement dataset.

In comparison with the MTL applied models, both models do not improve the performance of the classification task. Encoder information sharing is beneficial for both the classification and generation tasks, whereas the attention infor-

Proposed Method (MTL + SD + HCL)	Baseline
Input Articles with Attention Weight (Truncated) In 2011, Naka, the representative of Wantedly, founded our company alone. Because we first developed the web service (now "Wantedly"), we have used **Ruby on Rails** (...) Our development team is building valuable services with cooperation among engineers (...) Wantedly is at the height of its growth! We are recruiting **super geeks** to develop our "make work enjoyable" services with **Ruby on Rails** to achieve our mission of "Becoming the infrastructure for all workers" as a business SNS!	**Input Articles with Attention Weight (Truncated)** In 2011, Naka, the representative of Wantedly, founded our company alone. Because we first developed the web service (now "Wantedly"), we have used Ruby on Rails (...) **Our development team is building** valuable services with cooperation **among engineers** (...) Wantedly is at the height of its growth! We are recruiting super geeks to develop our "***make work enjoyable***" services with Ruby on Rails to achieve our mission of "Becoming the infrastructure for all workers" as a business SNS!
Generated Headline: We want a Ruby engineer to develop our own service!	**Generated Headline:** We want a new member to launch new businesses with us!
Generated Key Phrase: Ruby engineer	**Generated Key Phrase:** Engineer

Figure 5: An example of an input article, generated headlines and key phrases. Underlined words indicate that the attention weights of the headline generator between these words and the output words are high. **Blue colored words** indicate that the attention weights of these words are consistently high between the two decoders. In contrast, ***red colored italic words*** indicate that the attention weights of these words are high in the key phrase generator, but low in the headline generator, that is, inconsistency occurs between the two decoders. A Japanese-English translation is applied.

	ROUGE-1	ROUGE-2	ROUGE-L
Baseline	29.1	5.5	29.0
Proposed	53.7	19.2	53.4

Table 7: Comparison of ROUGE recall scores of generated key phrases against generated headlines for the job advertisement dataset. The proposed method significantly increases the word overlap between generated sentences.

mation sharing method is beneficial only for the generation task. This can be attributed to the asymmetry in the task settings. Generally, the generation task need to focus on a wider range of input words than the classification tasks. For the generation task, it is important that which words are focused on in the classification task. However, this is not always the case that the words focused on in the generation task are important for the classification task.

Analysis of word overlap between output sentences. To estimate the consistency between output sentences, we evaluate the word overlap between generated headlines and key phrases. Table 7 presents the ROUGE recall scores of the generated key phrases computed against generated headlines for the job advertisement dataset. Therefore, our proposed method generates significantly similar word outputs, resulting in improvement in the consistency between two sentences.

4.3 Example of Predicted Outputs

We visualize the attention weights of the decoders to assess the performance of the proposed method. The upper part of Figure 5 indicates an example of the attention weights of the headline and key phrase generators. The proposed method increases the consistency and decreases the inconsistency between the attention weights of the two decoders. With the proposed method, both decoders commonly focus on the word "Ruby on Rails," and thus both outputs consistently contain the word "Ruby." From this result, the proposed method, which avoids inconsistency, improves the quality of the outputs.

5 Conclusion

We introduced a novel multi-task learning method to maintain consistency among outputs. Hierarchical consistency loss was introduced to penalize inconsistency between two attention weights of decoders. We implemented a manual evaluation using crowd-sourcing, and the results indicates that our method generates more consistent outputs. An automatic evaluation showed that proposed method achieves the best ROUGE scores on both datasets. As a future work, we would like to explore whether our method is applicable to other tasks such as multi-task learning for object detection and image caption generation.

References

Antonios Anastasopoulos and David Chiang. 2018. Tied multitask learning for neural speech translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 82–91.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 484–494.

Kyunghyun Cho, Bart van Merrienboer, 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*, pages 1724–1734.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159.

Han Guo, Ramakanth Pasunuru, and Mohit Bansal. 2018. Soft layer-specific multi-task summarization with entailment and question generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, (Volume 1: Long Papers)*, pages 687–697.

Kazuma Hashimoto, Yoshimasa Tsuruoka, Richard Socher, et al. 2017. A joint many-task model: Growing a neural network for multiple NLP tasks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1923–1933.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems 28*, pages 1693–1701.

Wan-Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, Jing Tang, and Min Sun. 2018. A unified model for extractive and abstractive summarization using inconsistency loss. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 132–141.

Masaru Isonuma, Toru Fujino, Junichiro Mori, Yutaka Matsuo, and Ichiro Sakata. 2017. Extractive summarization using multi-task learning with document classification. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2101–2110.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*.

Eliyahu Kiperwasser and Miguel Ballesteros. 2018. Scheduled multi-task learning: From syntax to translation. *Transactions of the Association of Computational Linguistics*, 6:225–240.

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

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-task deep neural networks for natural language understanding. *arXiv preprint arXiv:1901.11504*.

Minh-Thang Luong, Quoc V Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2016. Multi-task sequence to sequence learning. In *Proceedings of 4th the International Conference on Learning Representation*.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.

Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep keyphrase generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 582–592.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290.

Eric W Noreen. 1989. *Computer-intensive methods for testing hypotheses*. A Wiley-Interscience Publication, New York: Wiley, 1989.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389.

Toshinori Sato, Taiichi Hashimoto, and Manabu Okumura. 2017. Implementation of a word segmentation dictionary called mecab-ipadic-NEologd and study on how to use it effectively for information retrieval. In *Proceedings of the 23th Annual Meeting of the Association for Natural Language Processing*.

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 (Volume 1: Long Papers), pages 1073–1083.

Shi-Qi Shen, Yan-Kai Lin, Cun-Chao Tu, Yu Zhao, Zhi-Yuan Liu, Mao-Song Sun, et al. 2017. Recent advances on neural headline generation. *Journal of Computer Science and Technology*, 32(4):768–784.

Tian Shi, Yaser Keneshloo, Naren Ramakrishnan, and Chandan K Reddy. 2018. Neural abstractive text summarization with sequence-to-sequence models. *arXiv preprint arXiv:1812.02303*.

Sho Takase, Jun Suzuki, Naoaki Okazaki, Tsutomu Hirao, and Masaaki Nagata. 2016. Neural headline generation on abstract meaning representation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1054–1059.

Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. 2017. From neural sentence summarization to headline generation: a coarse-to-fine approach. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 4109–4115.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 31*, pages 649–657.