Reducing Odd Generation from Neural Headline Generation

Shun Kiyono [1*] **Sho Takase** [2] **Jun Suzuki** [1,2,4]
Naoaki Okazaki [3] **Kentaro Inui** [1,4] **Masaaki Nagata** [2]
[1] Tohoku University [2] NTT Communication Science Laboratories
[3] Tokyo Institute of Technology [4] RIKEN Center for Advanced Intelligence Project
{kiyono,jun.suzuki,inui}@ecei.tohoku.ac.jp,
{takase.sho, nagata.masaaki}@lab.ntt.co.jp,
okazaki@c.titech.ac.jp

Abstract

The Encoder-Decoder model is widely used in natural language generation tasks. However, the model sometimes suffers from repeated redundant generation, misses important phrases, and includes irrelevant entities. Toward solving these problems we propose a novel source-side token prediction module. Our method jointly estimates the probability distributions over source and target vocabularies to capture the correspondence between source and target tokens. Experiments show that the proposed model outperforms the current state-of-the-art method in the headline generation task. We also show that our method can learn a reasonable token-wise correspondence without knowing any true alignment[1].

1 Introduction

The Encoder-Decoder model with the attention mechanism (EncDec) (Sutskever et al., 2014; Cho et al., 2014; Bahdanau et al., 2015; Luong et al., 2015) has been an epoch-making development that has led to great progress being made in many natural language generation tasks, such as machine translation (Bahdanau et al., 2015), dialog generation (Shang et al., 2015), and headline generation (Rush et al., 2015). Today, EncDec and its variants are widely used as the predominant baseline method in these tasks.

Unfortunately, as often discussed in the community, EncDec sometimes generates sentences with repeating phrases or completely irrelevant phrases and the reason for their generation cannot be interpreted intuitively. Moreover, EncDec also sometimes generates sentences that lack important phrases. We refer to these observations as the odd generation problem (*odd-gen*) in EncDec. The following table shows typical examples of *odd-gen* actually generated by a typical EncDec.

(1) Repeating Phrases	
Gold:	`duran duran group fashionable again`
EncDec:	`duran duran duran duran`
(2) Lack of Important Phrases	
Gold:	`graf says goodbye to tennis due to injuries`
EncDec:	`graf retires`
(3) Irrelevant Phrases	
Gold:	`u.s. troops take first position in serb-held bosnia`
EncDec:	`precede sarajevo`

This paper tackles for reducing the *odd-gen* in the task of abstractive summarization. In machine translation literature, coverage (Tu et al., 2016; Mi et al., 2016) and reconstruction (Tu et al., 2017) are promising extensions of EncDec to address the *odd-gen*. These models take advantage of the fact that machine translation is the *loss-less* generation (*lossless-gen*) task, where the semantic information of source- and target-side sequence is equivalent. However, as discussed in previous studies, abstractive summarization is a *lossy-compression* generation (*lossy-gen*) task. Here, the task is to delete certain semantic information from the source to generate target-side sequence.

*This work is a product of collaborative research program of Tohoku University and NTT Communication Science Laboratories.

[1]Our code for reproducing the experiments is available at https://github.com/butsugiri/UAM

Therefore the models such as the coverage and the reconstruction cannot work appropriately on abstractive summarization.

Recently, Zhou et al. (2017) proposed incorporating an additional gate for selecting an appropriate set of words from given source sentence. Moreover, Suzuki and Nagata (2017) introduced a module for estimating the upper-bound frequency of the target vocabulary given a source sentence. These methods essentially address individual parts of the *odd-gen* in *lossy-gen* tasks.

In contrast to the previous studies, we propose a novel approach that addresses the entire *odd-gen* in *lossy-gen* tasks. The basic idea underlying our method is to add an auxiliary module to EncDec for modeling the token-wise correspondence of the source and target, which includes drops of source-side tokens. We refer to our additional module as the Source-side Prediction Module (SPM). We add the SPM to the decoder output layer to directly estimate the correspondence during the training of EncDec.

We conduct experiments on a widely-used headline generation dataset (Rush et al., 2015) and evaluate the effectiveness of the proposed method. We show that the proposed method outperforms the current state-of-the-art method on this dataset. We also show that our method is able to learn a reasonable token-wise correspondence without knowing any true alignment, which may help reduce the *odd-gen*.

2 Lossy-compression Generation

We address the headline generation task introduced in Rush et al. (2015), which is a typical *lossy-gen* task. The source (input) is the first sentence of a news article, and the target (output) is the headline of the article. We say I and J represent the numbers of tokens in the source and target, respectively. An important assumption of the headline generation (*lossy-gen*) task is that the relation $I > J$ always holds, namely, the target must be shorter than the source. This implies that we need to optimally select salient concepts included in given source sentence. This selection indeed increases a difficulty of the headline generation for EncDec.

Note that it is an essentially difficult problem for EncDec to learn an appropriate paraphrasing of each concept in the source, which can be a main reason for irrelevant generation. In addition, EncDec also needs to manage the selection of concepts in the source; e.g., discarding an excessive number of concepts from the source would yield a headline that was too short, and utilizing the same concept multiple times may lead a redundant headline.

3 Encoder-Decoder Model with Attention Mechanism (EncDec)

This section briefly describes EncDec as the baseline model of our method[2]. To explain EncDec concisely, let us consider that the input of EncDec is a sequence of one-hot vectors $\boldsymbol{X}$ obtained from the given source-side sentence. Let $\boldsymbol{x}_i \in \{0,1\}^{V_s}$ represent the one-hot vector of the i-th token in $\boldsymbol{X}$, where V_s represents the number of instances (tokens) in the source-side vocabulary $\mathcal{V}_s$. We introduce $\boldsymbol{x}_{1:I}$ to represent $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_I)$ by a short notation, namely, $\boldsymbol{X} = \boldsymbol{x}_{1:I}$. Similarly, let $\boldsymbol{y}_j \in \{0,1\}^{V_t}$ represent the one-hot vector of the j-th token in target-side sequence $\boldsymbol{Y}$, where V_t is the number of instances (tokens) in the target-side vocabulary $\mathcal{V}_t$. Here, we define $\boldsymbol{Y}$ as always containing two additional one-hot vectors of special tokens $\langle bos \rangle$ for $\boldsymbol{y}_0$ and $\langle eos \rangle$ for $\boldsymbol{y}_{J+1}$. Thus, $\boldsymbol{Y} = \boldsymbol{y}_{0:J+1}$; its length is always $J+2$. Then, EncDec models the following conditional probability:

$$p(\boldsymbol{Y}|\boldsymbol{X}) = \prod_{j=1}^{J+1} p(\boldsymbol{y}_j|\boldsymbol{y}_{0:j-1}, \boldsymbol{X}). \quad (1)$$

EncDec encodes source one-hot vector sequence $\boldsymbol{x}_{1:I}$, and generates the hidden state sequence $\boldsymbol{h}_{1:I}$, where $\boldsymbol{h}_i \in \mathbb{R}^H$ for all i, and H is the size of the hidden state. Then, the decoder with the attention mechanism computes the vector $\boldsymbol{z}_j \in \mathbb{R}^H$ at every decoding time step j as:

$$\boldsymbol{z}_j = \mathrm{AttnDec}(\boldsymbol{y}_{j-1}, \boldsymbol{h}_{1:I}). \quad (2)$$

We apply RNN cells to both the encoder and decoder. Then, EncDec generates a target-side token based on the probability distribution $\boldsymbol{o}_j \in \mathbb{R}^{V_t}$ as:

$$\boldsymbol{o}_j = \mathrm{softmax}(\boldsymbol{W}_o \boldsymbol{z}_j + \boldsymbol{b}_o), \quad (3)$$

where $\boldsymbol{W}_o \in \mathbb{R}^{V_t \times H}$ is a parameter matrix and $\boldsymbol{b}_o \in \mathbb{R}^{V_t}$ is a bias term[3].

[2] Our model configuration follows EncDec described in Luong et al. (2015).

[3] For more detailed definitions of the encoder, decoder, and attention mechanism, see Appendices A, B and C.

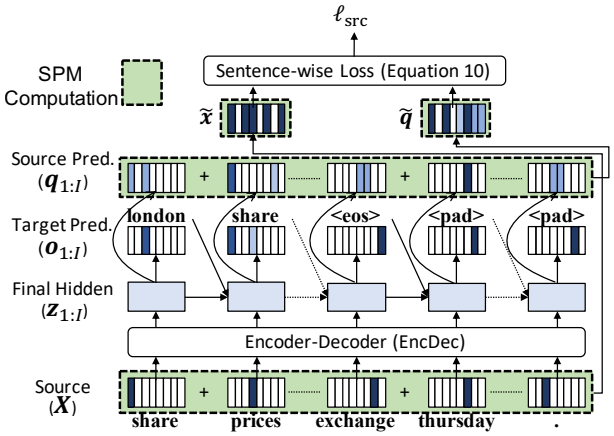


Figure 1: Overview of EncDec+SPM. The module inside the dashed rectangular box represents the SPM. The SPM predicts the probability distribution over the source vocabulary $\boldsymbol{q}_j$ at each time step j. After predicting all the time steps, the SPM compares the sum of the predictions $\tilde{\boldsymbol{q}}$ with the sum of the source-side tokens $\tilde{\boldsymbol{x}}$ as an objective function ℓ_{src}.

To train EncDec, let $\mathcal{D}$ be training data for headline generation that consists of source-headline sentence pairs. Let θ represent all parameters in EncDec. Our goal is to find the optimal parameter set $\hat{\theta}$ that minimizes the following objective function $G_1(\theta)$ for the given training data $\mathcal{D}$:

$$G_1(\theta) = \frac{1}{|\mathcal{D}|} \sum_{(\boldsymbol{X},\boldsymbol{Y})\in\mathcal{D}} \ell_{\mathrm{trg}}(\boldsymbol{Y}, \boldsymbol{X}, \theta),$$
$$\ell_{\mathrm{trg}}(\boldsymbol{Y}, \boldsymbol{X}, \theta) = -\log\Big(p(\boldsymbol{Y}|\boldsymbol{X}, \theta)\Big). \quad (4)$$

Since $\boldsymbol{o}_j$ for each j is a vector representation of the probabilities of $p(\hat{\boldsymbol{y}}|\boldsymbol{y}_{0:j-1}, \boldsymbol{X}, \theta)$ over the target vocabularies $\hat{\boldsymbol{y}} \in \mathcal{V}_t$, we can calculate ℓ_{trg} as:

$$\ell_{\mathrm{trg}}(\boldsymbol{Y}, \boldsymbol{X}, \theta) = -\sum_{j=1}^{J+1} \boldsymbol{y}_j^{\top} \cdot \log\big(\boldsymbol{o}_j\big). \quad (5)$$

In the inference step, we use the trained parameters to search for the best target sequence. We use beam search to find the target sequence that maximizes the product of the conditional probabilities as described in Equation 1.

4 Proposed Method: Source Prediction Module (SPM)

In Section 2, we assumed that the selection of concepts in the source is an essential part of the *odd-gen*. Thus, our basic idea is to extend EncDec that can manage the status of concept utilization during headline generation. More precisely, instead of directly managing concepts since they are not well-defined, We consider to model token-wise correspondence of the source and target, including the information of source-side tokens that cannot be aligned to any target-side tokens.

Figure 1 overviews the proposed method, SPM. During the training process of EncDec, the decoder estimates the probability distribution over source-side vocabulary, which is $\boldsymbol{q}_j \in \mathbb{R}^{V_s}$, in addition to that of the target-side vocabulary, $\boldsymbol{o}_j \in \mathbb{R}^{V_t}$, for each time step j. Note that the decoder continues to estimate the distributions up to source sequence length I regardless of target sequence length J. Here, we introduce a special token $\langle pad \rangle$ in the target-side vocabulary, and assume that $\langle pad \rangle$ is repeatedly generated after finishing the generation of all target-side tokens as correct target tokens. This means that we always assume that the numbers of tokens in the source and target is the same, and thus, our method allows to put one-to-one correspondence into practice in the *lossy-gen* task. In this way, EncDec can directly model token-wise correspondence of source- and target-side tokens on the decoder output layer, which includes the information of *unaligned* source-side tokens by alignment to $\langle pad \rangle$.

Unfortunately, standard headline generation datasets have no information of true one-to-one alignments between source- and target-side tokens. Thus, we develop a novel method for training token-wise correspondence model that takes unsupervised learning approach. Specifically, we minimize sentence-level loss instead of token-wise alignment loss. We describe the details in the following sections.

4.1 Model Definition

In Figure 1, the module inside the dashed line represents the SPM. First, the SPM calculates a probability distribution over the source vocabulary $\boldsymbol{q}_j \in \mathbb{R}^{V_s}$ at each time step j in the decoding process by using the following equation:

$$\boldsymbol{q}_j = \mathrm{softmax}(\boldsymbol{W}_q \boldsymbol{z}_j + \boldsymbol{b}_q), \quad (6)$$

where $\boldsymbol{W}_q \in \mathbb{R}^{V_s \times H}$ is a parameter matrix like $\boldsymbol{W}_o$ in Equation 3, and $\boldsymbol{b}_q \in \mathbb{R}^{V_s}$ is a bias term. As described in Section 3, EncDec calculates a probability

distribution over the target vocabulary $\boldsymbol{o}_j$ from $\boldsymbol{z}_j$. Therefore, EncDec with the SPM jointly estimates the probability distributions over the source and target vocabularies from the same vector $\boldsymbol{z}_j$.

Next, we define $\boldsymbol{Y}'$ as a concatenation of the one-hot vectors of the target-side sequence $\boldsymbol{Y}$ and those of the special token $\langle pad \rangle$ of length $I-(J+1)$. Here, $\boldsymbol{y}_{J+1}$ is a one-hot vector of $\langle eos \rangle$, and $\boldsymbol{y}_j$ for each $j \in \{J+2, \ldots, I\}$ is a one-hot vector of $\langle pad \rangle$. We define $\boldsymbol{Y}' = \boldsymbol{Y}$ if and only if $J+1 = I$. Note that the length of $\boldsymbol{Y}'$ is always no shorter than that of $\boldsymbol{Y}$, that is, $|\boldsymbol{Y}'| \geq |\boldsymbol{Y}|$ since headline generation always assumes $I > J$ as described in Section 2.

Let $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{q}}$ be the sum of all one-hot vectors in source sequence $\boldsymbol{x}_{1:I}$ and all prediction of the SPM $\boldsymbol{q}_{1:I}$, respectively; that is, $\tilde{\boldsymbol{x}} = \sum_{i=1}^{I} \boldsymbol{x}_i$ and $\tilde{\boldsymbol{q}} = \sum_{j=1}^{I} \boldsymbol{q}_j$. Note that $\tilde{\boldsymbol{x}}$ is a vector representation of the occurrence (or bag-of-words representation) of each source-side vocabulary appearing in the given source sequence.

EncDec with the SPM models the following conditional probability:

$$p(\boldsymbol{Y}', \tilde{\boldsymbol{x}}|\boldsymbol{X}) = p(\tilde{\boldsymbol{x}}|\boldsymbol{Y}', \boldsymbol{X})p(\boldsymbol{Y}'|\boldsymbol{X}). \quad (7)$$

We define $p(\boldsymbol{Y}'|\boldsymbol{X})$ as follows:

$$p(\boldsymbol{Y}'|\boldsymbol{X}) = \prod_{j=1}^{I} p(\boldsymbol{y}_j|\boldsymbol{y}_{0:j-1}, \boldsymbol{X}), \quad (8)$$

which is identical to $p(\boldsymbol{Y}|\boldsymbol{X})$ in Equation 1 except for substituting I for J to model the probabilities of $\langle pad \rangle$ that appear from $j = I-(J+1)$ to $j = I$. Next, we define $p(\tilde{\boldsymbol{x}}|\boldsymbol{Y}', \boldsymbol{X})$ as follows:

$$p(\tilde{\boldsymbol{x}}|\boldsymbol{Y}', \boldsymbol{X}) = \frac{1}{Z} \exp\left(\frac{-\|\tilde{\boldsymbol{q}} - \tilde{\boldsymbol{x}}\|_2^2}{C}\right), \quad (9)$$

where Z is a normalization term, and C is a hyper-parameter that controls the sensitivity of the distribution.

4.2 Training and Inference of SPM

Training Let γ represent the parameter set of SPM. Then, we define the loss function for SPM as

$$\ell_{\mathrm{src}}(\tilde{\boldsymbol{x}}, \boldsymbol{X}, \boldsymbol{Y}', \gamma, \theta) = -\log\Big(p(\tilde{\boldsymbol{x}}|\boldsymbol{Y}', \boldsymbol{X}, \gamma, \theta)\Big).$$

From Equation 9, we can derive ℓ_{src} as

$$\ell_{\mathrm{src}}(\tilde{\boldsymbol{x}}, \boldsymbol{X}, \boldsymbol{Y}', \gamma, \theta) = \frac{1}{C}\|\tilde{\boldsymbol{q}} - \tilde{\boldsymbol{x}}\|_2^2 + \log(Z). \quad (10)$$

We can discard the second term on the RHS, that is $\log(Z)$, since this is independent of γ and θ.

Here, we regard the sum of ℓ_{trg} and ℓ_{src} as an objective loss function of multi-task training. Formally, we train the SPM with EncDec by minimizing the following objective function G_2:

$$G_2(\theta, \gamma) = \frac{1}{|\mathcal{D}|}\sum_{(\boldsymbol{X},\boldsymbol{Y})\in\mathcal{D}}\Big(\ell_{\mathrm{trg}}(\boldsymbol{Y}', \boldsymbol{X}, \theta) + \ell_{\mathrm{src}}(\tilde{\boldsymbol{x}}, \boldsymbol{X}, \boldsymbol{Y}', \gamma, \theta)\Big) \quad (11)$$

Inference In the inference time, the goal is only to search for the best target sequence. Thus, we do not need to compute SPM during the inference. Similarly, it is also unnecessary to produce $\langle pad \rangle$ after generating $\langle eos \rangle$. Thus, the actual computation cost of our method for the standard evaluation is exactly the same as that of the base EncDec.

5 Experiment

5.1 Settings

Dataset The origin of the headline generation dataset used in our experiments is identical to that used in Rush et al. (2015), namely, the dataset consists of pairs of the first sentence of each article and its headline from the annotated English Gigaword corpus (Napoles et al., 2012).

We slightly changed the data preparation procedure to achieve a more realistic and reasonable evaluation since the widely-used provided evaluation dataset already contains $\langle unk \rangle$, which is a replacement of all low frequency words. This is because the data preprocessing script provided by Rush et al. (2015)[4] automatically replaces low frequency words with $\langle unk \rangle$ [5]. To penalize $\langle unk \rangle$ in system outputs during the evaluation, we removed $\langle unk \rangle$ replacement procedure from the preprocessing script. We believe this is more realistic evaluation setting.

Rush et al. (2015) defined the training, validation and test split, which contain approximately 3.8M, 200K and 400K source-headline pairs, respectively. We used the entire training split for training as in

[4] https://github.com/facebookarchive/NAMAS.

[5] In a personal communication with the first author of Zhou et al. (2017), we found that their model decodes $\langle unk \rangle$ in the same form as it appears in the test set, and $\langle unk \rangle$ had a positive effect on the final performance of the model.

the previous studies. We randomly sampled test data and validation data from the validation split since we found that the test split contains many noisy instances. Finally, our validation and test data consist of 8K and 10K source-headline pairs, respectively. Note that they are relatively large compared with the previously used datasets, and they do not contain ⟨*unk*⟩.

We also evaluated our experiments on the test data used in the previous studies. To the best of our knowledge, two test sets from the Gigaword are publicly available by Rush et al. (2015)[6] and Zhou et al. (2017)[7]. Here, both test sets contain ⟨*unk*⟩ [8].

Evaluation Metric We evaluated the performance by ROUGE-1 (RG-1), ROUGE-2 (RG-2) and ROUGE-L (RG-L)[9]. We report the F1 value as given in a previous study[10]. We computed the scores with the official ROUGE script (version 1.5.5).

Comparative Methods To investigate the effectiveness of the SPM, we evaluate the performance of the EncDec with the SPM. In addition, we investigate whether the SPM improves the performance of the state-of-the-art method: EncDec+sGate. Thus, we compare the following methods on the same training setting.

EncDec This is the implementation of the base model explained in Section 3.

EncDec+sGate To reproduce the state-of-the-art method proposed by Zhou et al. (2017), we combined our re-implemented selective gate (sGate) with the encoder of EncDec.

EncDec+SPM We combined the SPM with the EncDec as explained in Section 4.

EncDec+sGate+SPM This is the combination of the SPM with the EncDec+sGate.

Implementation Details Table 1 summarizes hyper-parameters and model configurations. We selected the settings commonly-used in the previous studies, *e.g.*, (Rush et al., 2015; Nallapati et al., 2016; Suzuki and Nagata, 2017).

We constructed the vocabulary set using Byte-Pair-Encoding[11] (BPE) (Sennrich et al., 2016) to handle low frequency words, as it is now a common practice in neural machine translation. The BPE merge operations are jointly learned from the source and the target. We set the number of the BPE merge operations at $5,000$. We used the same vocabulary set for both the source $\mathcal{V}_s$ and the target $\mathcal{V}_t$.

Source Vocab. Size V_s	5131
Target Vocab. Size V_t	5131
Word Embed. Size D	200
Hidden State Size H	400
RNN Cell	Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997)
Encoder RNN Unit	2-layer bidirectional-LSTM
Decoder RNN Unit	2-layer LSTM with attention (Luong et al., 2015)
Optimizer	Adam (Kingma and Ba, 2015)
Initial Learning Rate	0.001
Learning Rate Decay	0.5 for each epoch (after epoch 9)
Weight C of ℓ_{src}	10
Mini-batch Size	256 (shuffled at each epoch)
Gradient Clipping	5
Stopping Criterion	Max 15 epochs with early stopping
Regularization	Dropout (rate 0.3)
Beam Search	Beam size 20 with the length normalization

Table 1: Configurations used in our experiments

5.2 Results

Table 2 summarizes results for all test data. The table consists of three parts split by horizontal lines. The top and middle rows show the results on our training procedure, and the bottom row shows the results reported in previous studies. Note that the top and middle rows are not directly comparable to the bottom row due to the differences in preprocessing and vocabulary settings.

The top row of Table 2 shows that EncDec+SPM outperformed both EncDec and EncDec+sGate. This result indicates that the SPM can improve the performance of EncDec. Moreover, it is noteworthy that EncDec+sGate+SPM achieved the best performance in all metrics even though EncDec+sGate consists of essentially the same architecture as the current

[6] https://github.com/harvardnlp/sent-summary

[7] https://res.qyzhou.me

[8] We summarize the details of the dataset in Appendix D

[9] We restored sub-words to the standard token split for the evaluation.

[10] ROUGE script option is: "-n2 -m -w 1.2"

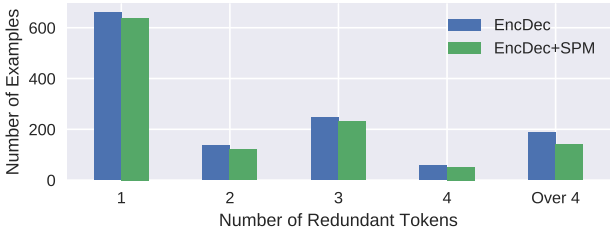
[11] https://github.com/rsennrich/subword-nmt

Models	Gigaword Test (Ours)			Gigaword Test (Rush)			Gigaword Test (Zhou)		
	RG-1	RG-2	RG-L	RG-1	RG-2	RG-L	RG-1	RG-2	RG-L
EncDec	45.74	23.80	42.95	34.52	16.77	32.19	45.62	24.26	42.87
EncDec+sGate (our impl. of SEASS)	45.98	24.17	43.16	35.00	17.24	32.72	45.96	24.63	43.18
EncDec+SPM†	46.18	24.34	43.35	35.17	17.07	32.75	46.21	24.78	43.27
EncDec+sGate+SPM†	**46.41**	**24.58**	**43.59**	**35.79**	**17.84**	**33.34**	**46.34**	**24.85**	**43.49**
EncDec+sGate+SPM (5 Ensemble)†	**47.16**	**25.34**	**44.34**	**36.23**	**18.11**	**33.79**	**47.39**	**25.89**	**44.41**
ABS (Rush et al., 2015)	-	-	-	29.55	11.32	26.42	37.41	15.87	34.70
SEASS (Zhou et al., 2017)	-	-	-	36.15	17.54	33.63	46.86	24.58	43.53
DRGD (Li et al., 2017)	-	-	-	36.27	17.57	33.62	-	-	-
WFE (Suzuki and Nagata, 2017)	-	-	-	36.30	17.31	33.88	-	-	-

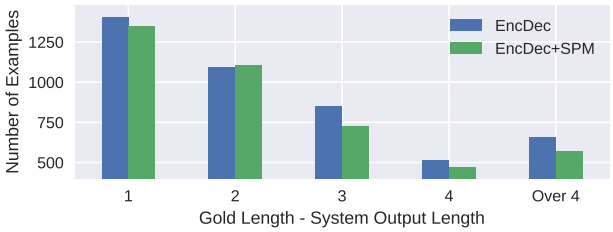
Table 2: Full length ROUGE F1 evaluation results. The top and middle rows show the results on our evaluation setting. † is the proposed model. The bottom row shows published scores reported in previous studies. Note that (1) SEASS consists of essentially the same architecture as our implemented EncDec+sGate, and (2) the top and middle rows are not directly comparable to the bottom row due to differences in preprocessing and vocabulary settings. (see discussions in Section 5.2).

state-of-the-art model, *i.e.*, SEASS.

The bottom row of Table 2 shows the results of previous methods. They often obtained higher ROUGE scores than our models in Gigaword Test (Rush) and Gigaword Test (Zhou). However, this does not immediately imply that our method is inferior to the previous methods. This result is basically derived from the inconsistency of the vocabulary. In detail, our training data does not contain $\langle unk \rangle$ because we adopted the BPE to construct the vocabulary. Thus, our experimental setting is severer than that of previous studies with the presence of $\langle unk \rangle$ in the datasets of Gigaword Test (Rush) and Gigaword Test (Zhou). This is also demonstrated by the fact that EncDec+sGate obtained a lower score than those reported in the paper of SEASS, which has the same model architecture as EncDec+sGate.



(a) Repeating Phrases



(b) Lack of Important Phrases

Figure 2: Comparison between EncDec and EncDec+SPM on the number of sentences that potentially contain the *odd-gen*. The smaller examples mean reduction of the *odd-gen*.

6 Discussion

The motivation of the SPM is to suppress *odd-gen* by enabling a one-to-one correspondence between the source and the target. Thus, in this section, we investigate whether the SPM reduces *odd-gen* in comparison to EncDec.

6.1 Does SPM Reduce *odd-gen*?

For a useful quantitative analysis, we should compute the statistics of generated sentences containing *odd-gen*. However, it is hard to detect *odd-gen* correctly. Instead, we determine a pseudo count of each type of *odd-gen* as follows.

Repeating phrases We assume that a model causes repeating phrases if the model outputs the same token more than once. Therefore, we compute the frequency of tokens that occur more than once in the generated headlines. However, some phrases might occur more than once in the gold data. To address this case, we subtract the frequency of tokens

(1) Repeating Phrases			
Gold:	duran duran group fashionable again	Gold:	community college considers building $ ## million technology
EncDec:	duran duran duran duran	EncDec:	college college colleges learn to get ideas for tech center
EncDec+SPM:	duran duran fashionably cool once again	EncDec+SPM:	l.a. community college officials say they 'll get ideas
(2) Lack of Important Phrases			
Gold:	graf says goodbye to tennis due to injuries	Gold:	new york 's primary is most suspenseful of super tuesday races
EncDec:	graf retires	EncDec:	n.y.
EncDec+SPM:	german tennis legend steffi graf retires	EncDec+SPM:	new york primary enters most suspenseful of super tuesday contests
(3) Irrelevant Phrases			
Gold:	u.s. troops take first position in serb-held bosnia	Gold:	northridge hopes confidence does n't wane
EncDec:	precede sarajevo	EncDec:	csun 's csun
EncDec+SPM:	u.s. troops set up first post in bosnian countryside	EncDec+SPM:	northridge tries to win northridge men 's basketball team

Figure 3: Examples of generated summaries. "Gold" indicates the reference headline. The proposed EncDec+SPM model successfully reduced *odd-gen*.

in the reference headline from the above calculation result. The result of this subtraction is taken to be the number of repeating phrases in each generated headline.

Lack of important phrases We assume generated headline which is shorter than the gold omits one or more important phrase. Thus, we compute the difference in gold headline length and the generated headline length.

Irrelevant phrases We consider that improvements in ROUGE scores indicate a reduction in irrelevant phrases because we believe that ROUGE penalizes irrelevant phrases.

Figure 2 shows the number of repeating phrases and lack of important phrases in Gigaword Test (Ours). This figure indicates that EncDec+SPM reduces the *odd-gen* in comparison to EncDec. Thus, we consider the SPM reduced *odd-gen*. Figure 3 shows sampled headlines actually generated by EncDec and EncDec+SPM. It is clear that the outputs of EncDec contain *odd-gen* while those of the EncDec+SPM do not. These examples also demonstrate that SPM successfully reduces *odd-gen*.

6.2 Visualizing SPM and Attention

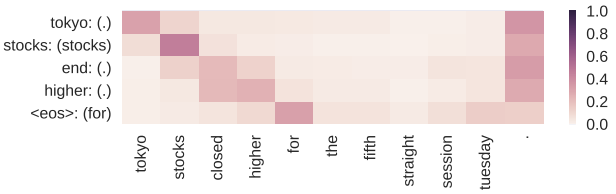
We visualize the prediction of the SPM and the attention distribution to see the acquired token-wise correspondence between the source and the target. Specifically, we feed the source-target pair $(\boldsymbol{X}, \boldsymbol{Y})$ to EncDec and EncDec+SPM, and then collect the source-side prediction $(\boldsymbol{q}_{1:I})$ of EncDec+SPM and the attention distribution $(\boldsymbol{\alpha}_{1:J})$ of EncDec[12]. For source-side prediction, we extracted the probability of each token $\boldsymbol{x}_i \in \boldsymbol{X}$ from $\boldsymbol{q}_j, j \in \{1, \ldots, I\}$.

Figure 4 shows an example of the heat map[13]. We used Gigaword Test (Ours) as the input. The brackets in the y-axis represent the source-side tokens that are aligned with target-side tokens. We selected the aligned tokens in the following manner: For the attention (Figure 4a), we select the token with the largest attention value. For the SPM (Figure 4b), we select the token with the largest probability over the whole vocabulary $\mathcal{V}_s$.

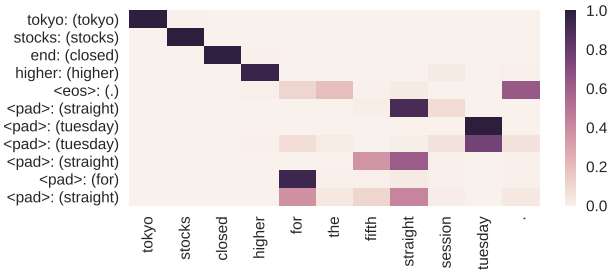
Figure 4a indicates that most of the attention distribution is concentrated at the end of the sentence. As a result, attention provides poor token-wise correspondence between the source and the target. For example, target-side tokens "tokyo" and "end" are both aligned with the source-side sentence period. In contrast, Figure 4b shows that the SPM captures the correspondence between the source and the target. The source sequence "tokyo stocks closed higher" is successfully aligned with the target "tokyo stocks end higher". Moreover, the SPM aligned unimportant to-

[12]For details of the attention mechanism, see Appendix C.

[13]For more visualizations, see Appendix E



(a) Attention distribution of EncDec



(b) Source-side prediction of EncDec+SPM

Figure 4: Visualization of EncDec and EncDec+SPM. The x-axis and y-axis of the figure correspond to the source and the target sequence respectively. Tokens in the brackets represent source-side tokens that are aligned with target-side tokens at that time step.

kens for the headline such as "straight" and "tuesday" with $\langle pad \rangle$ tokens. Thus, this example suggests that the SPM achieves better token-wise correspondence than attention. It is noteworthy that the SPM captured a one-to-one correspondence even though we trained it without correct alignment information.

7 Related Work

In the field of neural machine translation, several methods have been proposed to solve the *odd-gen*. The coverage model (Mi et al., 2016; Tu et al., 2016) forces the decoder to attend to every part of the source sequence to translate all semantic information in the source. The reconstructor (Tu et al., 2017) trains the translation model from the target to the source. Moreover, Weng et al. (2017) proposed a method to predict the untranslated words from the decoder at each time step. These methods aim to convert all contents in the source into the target, since machine translation is a *lossless-gen* task. In contrast, our proposal, SPM, models both paraphrasing and discarding to reduce the *odd-gen* in the *lossy-gen* task.

We focused on headline generation which is a well-known *lossy-gen* task. Recent studies have actively applied the EncDec to this task (Rush et al., 2015; Chopra et al., 2016; Nallapati et al., 2016). For the headline generation task, Zhou et al. (2017) and Suzuki and Nagata (2017) tackled a part of the *odd-gen*. Zhou et al. (2017) incorporated an additional gate (sGate) into the encoder to select appropriate words from the source. Suzuki and Nagata (2017) proposed a frequency estimation module to reduce the repeating phrases. Our motivation is similar to theirs, but our goal is to solve all *odd-gen* components. In addition, we can combine these approaches with the proposed method. In fact, we showed in Section 5.2 that the SPM can improve the performance of sGate with EncDec.

Apart from tackling *odd-gen*, some studies proposed methods to improve the performance of the headline generation task. Takase et al. (2016) incorporated AMR (Banarescu et al., 2013) into the encoder to use the syntactic and semantic information of the source. Nallapati et al. (2016) also encoded additional information of the source such as TF-IDF and named entities. Li et al. (2017) modeled the typical structure of a headline, such as "Who Action What" with a variational auto-encoder. These approaches improve the performance of headline generation, but it is unclear that they can reduce *odd-gen*.

8 Conclusion

In this paper, we introduced an approach for reducing the *odd-gen* in the *lossy-gen* task. The proposal, SPM, learns to predict the one-to-one correspondence of tokens in the source and the target. Experiments on the headline generation task showed that the SPM improved the performance of typical EncDec, and outperformed the current state-of-the-art model. Furthermore, we demonstrated that the SPM reduced the *odd-gen*. In addition, SPM obtained token-wise correspondence between the source and the target without any alignment data.

Acknowledgment

We are grateful to anonymous reviewers for their insightful comments. We thank Sosuke Kobayashi for providing helpful comments. We also thank Qingyu Zhou for providing a dataset and information for a fair comparison.

References

[Bahdanau et al.2015] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015)*.

[Banarescu et al.2013] Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for Sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186.

[Cho et al.2014] 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 (EMNLP 2014)*, pages 1724–1734.

[Chopra et al.2016] Sumit Chopra, Michael Auli, and Alexander M. Rush. 2016. Abstractive Sentence Summarization with Attentive Recurrent Neural Networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2016)*, pages 93–98.

[Hochreiter and Schmidhuber1997] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

[Kingma and Ba2015] Diederik Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015)*.

[Li et al.2017] Piji Li, Wai Lam, Lidong Bing, and Zihao Wang. 2017. Deep Recurrent Generative Decoder for Abstractive Text Summarization. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, pages 2081–2090.

[Luong et al.2015] 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 (EMNLP 2015)*, pages 1412–1421.

[Mi et al.2016] Haitao Mi, Baskaran Sankaran, Zhiguo Wang, and Abe Ittycheriah. 2016. Coverage Embedding Models for Neural Machine Translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, pages 955–960.

[Nallapati et al.2016] 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.

[Napoles et al.2012] Courtney Napoles, Matthew Gormley, and Benjamin Van Durme. 2012. Annotated Gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, AKBC-WEKEX '12, pages 95–100.

[Rush et al.2015] 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 (EMNLP 2015)*, pages 379–389.

[Sennrich et al.2016] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, pages 1715–1725.

[Shang et al.2015] Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural Responding Machine for Short-Text Conversation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL & IJCNLP 2015)*, pages 1577–1586, July.

[Sutskever et al.2014] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. In *Advances in Neural Information Processing Systems 27 (NIPS 2014)*, pages 3104–3112.

[Suzuki and Nagata2017] Jun Suzuki and Masaaki Nagata. 2017. Cutting-off Redundant Repeating Generations for Neural Abstractive Summarization. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2017)*, pages 291–297.

[Takase et al.2016] 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 (EMNLP 2016)*, pages 1054–1059.

[Tu et al.2016] Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling Coverage for Neural Machine Translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, pages 76–85.

[Tu et al.2017] Zhaopeng Tu, Yang Liu, Lifeng Shang, Xiaohua Liu, and Hang Li. 2017. Neural Machine Translation with Reconstruction. In *Thirty-First AAAI Conference on Artificial Intelligence (AAAI 2017)*, pages 3097–3103.

[Weng et al.2017] Rongxiang Weng, Shujian Huang, Zaixiang Zheng, Xinyu Dai, and Jiajun Chen. 2017. Neural Machine Translation with Word Predictions. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, pages 136–145.

[Zhou et al.2017] Qingyu Zhou, Nan Yang, Furu Wei, and Ming Zhou. 2017. Selective Encoding for Abstractive Sentence Summarization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, pages 1095–1104.

A Baseline Model Encoder

We employ bidirectional RNN (BiRNN) as the encoder of the baseline model. BiRNN is composed of two separate RNNs for forward ($\overrightarrow{\mathrm{RNN}}_{\mathrm{src}}$) and backward ($\overleftarrow{\mathrm{RNN}}_{\mathrm{src}}$) directions. The forward RNN reads the source sequence $\boldsymbol{X}$ from left to right order and constructs hidden states $(\vec{\boldsymbol{h}}_1, \ldots, \vec{\boldsymbol{h}}_I)$. Similarly, the backward RNN reads the input in the reverse order to obtain another sequence of hidden states $(\overleftarrow{\boldsymbol{h}}_1, \ldots, \overleftarrow{\boldsymbol{h}}_I)$. Finally, we take a summation of the hidden states of each direction to construct the final representation of the source sequence $(\boldsymbol{h}_1, \ldots, \boldsymbol{h}_I)$.

Concretely, for given time step i, the representation $\boldsymbol{h}_i$ is constructed as follows:

$$\vec{\boldsymbol{h}}_i = \overrightarrow{\mathrm{RNN}}_{\mathrm{src}}(\boldsymbol{E}_s \boldsymbol{x}_i, \vec{\boldsymbol{h}}_{i-1}), \tag{12}$$

$$\overleftarrow{\boldsymbol{h}}_i = \overleftarrow{\mathrm{RNN}}_{\mathrm{src}}(\boldsymbol{E}_s \boldsymbol{x}_i, \overleftarrow{\boldsymbol{h}}_{i+1}), \tag{13}$$

$$\boldsymbol{h}_i = \vec{\boldsymbol{h}}_i + \overleftarrow{\boldsymbol{h}}_i \tag{14}$$

where $\boldsymbol{E}_s \in \mathbb{R}^{D \times V_s}$ denotes the word embedding matrix of the source-side, and D denotes the size of word embedding.

B Baseline Model Decoder

The baseline model AttnDec is composed of the decoder and the attention mechanism. Here, the decoder is the unidirectional RNN with the input-feeding approach (Luong et al., 2015). Concretely, decoder RNN takes the output of the previous time step $\boldsymbol{y}_{j-1}$, decoder hidden state $\vec{\boldsymbol{z}}_{j-1}$ and final hidden state $\boldsymbol{z}_{j-1}$ and derives the hidden state of current time step $\boldsymbol{z}_j$:

$$\vec{\boldsymbol{z}}_j = \overrightarrow{\mathrm{RNN}}_{\mathrm{trg}}(\boldsymbol{E}_t \boldsymbol{y}_{j-1}, \boldsymbol{z}_{j-1}, \vec{\boldsymbol{z}}_{j-1}), \tag{15}$$

$$\vec{\boldsymbol{z}}_0 = \vec{\boldsymbol{h}}_I + \overleftarrow{\boldsymbol{h}}_1 \tag{16}$$

where $\boldsymbol{E}_t \in \mathbb{R}^{D \times V_t}$ denotes the word embedding matrix of the decoder. Here, $\boldsymbol{z}_0$ is defined as a zero vector.

C Baseline Model Attention Mechanism

The attention architecture of the baseline model is the same as the *Global Attention* model proposed by Luong et al. (2015). Attention is responsible for constructing the final hidden state $\boldsymbol{z}_j$ from the decoder hidden state $\vec{\boldsymbol{z}}_j$ and encoder hidden states $(\boldsymbol{h}_1, \ldots, \boldsymbol{h}_I)$.

	use $\langle unk \rangle$?	size	#.ref	source (split)
Training	No	3,778,230	1	Giga (train)
Validation	No	8,000	1	Giga (valid)
Test (ours)	No	10,000	1	Giga (valid)
Test (Rush)	Yes	1,951	1	Giga (test)
Test (Zhou)	Yes	2,000	1	Giga (valid)

Table 3: Characteristics of each dataset used in our experiments

First, the model computes the attention distribution $\boldsymbol{\alpha}_j \in \mathbb{R}^I$ from the decoder hidden state $\vec{\boldsymbol{z}}_j$ and encoder hidden states $(\boldsymbol{h}_1, \ldots, \boldsymbol{h}_I)$. From among three attention scoring functions proposed in Luong et al. (2015), we employ *general* function. This function calculates the attention score in bilinear form. Specifically, the attention score between the i-th source hidden state and the j-th decoder hidden state is computed by the following equation:

$$\boldsymbol{\alpha}_j[i] = \frac{\exp(\boldsymbol{h}_i^\top \boldsymbol{W}_\alpha \vec{\boldsymbol{z}}_j)}{\sum_{i=1}^{I} \exp(\boldsymbol{h}_i^\top \boldsymbol{W}_\alpha \vec{\boldsymbol{z}}_j)} \tag{17}$$

where $\boldsymbol{W}_\alpha \in \mathbb{R}^{H \times H}$ is a parameter matrix, and $\boldsymbol{\alpha}_j[i]$ denotes i-th element of $\boldsymbol{\alpha}_j$.

$\boldsymbol{\alpha}_j$ is then used for collecting the source-side information that is relevant for predicting the target token. This is done by taking the weighted sum on the encoder hidden states:

$$\boldsymbol{c}_j = \sum_{i=1}^{I} \boldsymbol{\alpha}_j[i] \boldsymbol{h}_i \tag{18}$$

Finally, the source-side information is mixed with the decoder hidden state to derive final hidden state $\boldsymbol{z}_j$. Concretely, the context vector $\boldsymbol{c}_j$ is concatenated with $\vec{\boldsymbol{z}}_j$ to form vector $\boldsymbol{u}_j \in \mathbb{R}^{2H}$. $\boldsymbol{u}_j$ is then fed into a single fully-connected layer with $\tanh$ nonlinearity:

$$\boldsymbol{z}_j = \tanh(\boldsymbol{W}_s \boldsymbol{u}_j) \tag{19}$$

where $\boldsymbol{W}_s \in \mathbb{R}^{H \times 2H}$ is a parameter matrix.

D Dataset Summary

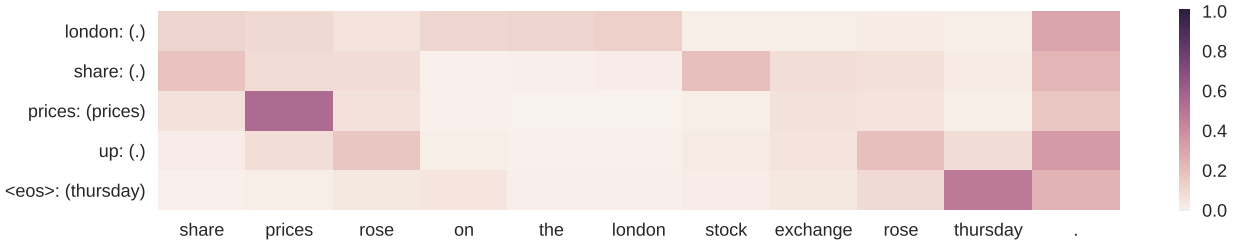
Table 3 summarizes the characteristics of each dataset used in our experiments.

E Extra Visualizations of SPM and Attention

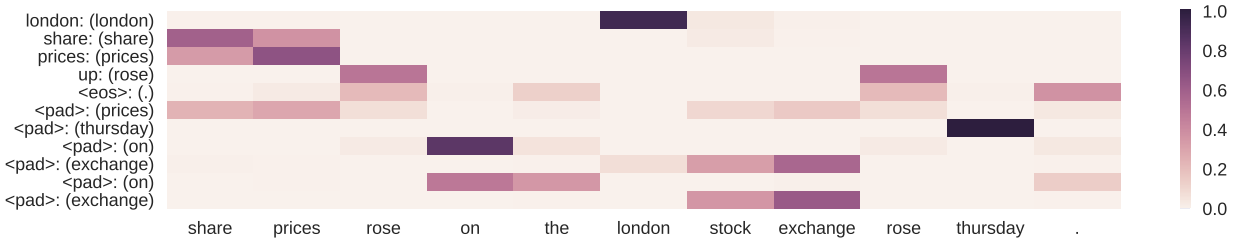
Figures 5, 6 and 7 are additional visualizations of SPM and attention. We created each figure using the procedure described in Section 6.2.

F Obtained Alignments

We analyzed source-side prediction to investigate the alignment acquired by SPM. We randomly sampled 500 source-target pairs from Gigaword Test (Ours), and fed them to EncDec+SPM. For each decoding time step j, we created the alignment pair by comparing the target-side token $\boldsymbol{y}_j$ with the token with the highest probability over the source-side probability distribution $\boldsymbol{q}_j$. Table 4 summarizes some examples of the obtained alignments. The table shows that the SPM aligns various types of word pairs, such as verb inflection and paraphrasing to the shorter form.



(a) Attention distribution of EncDec

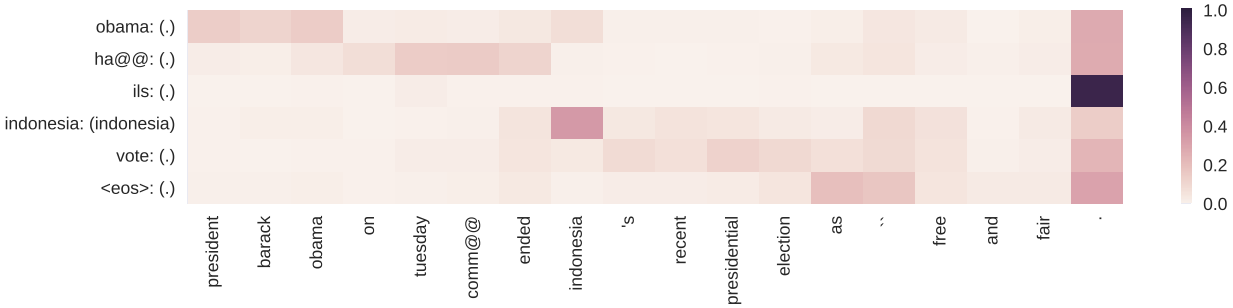


(b) Source-side prediction of EncDec+SPM

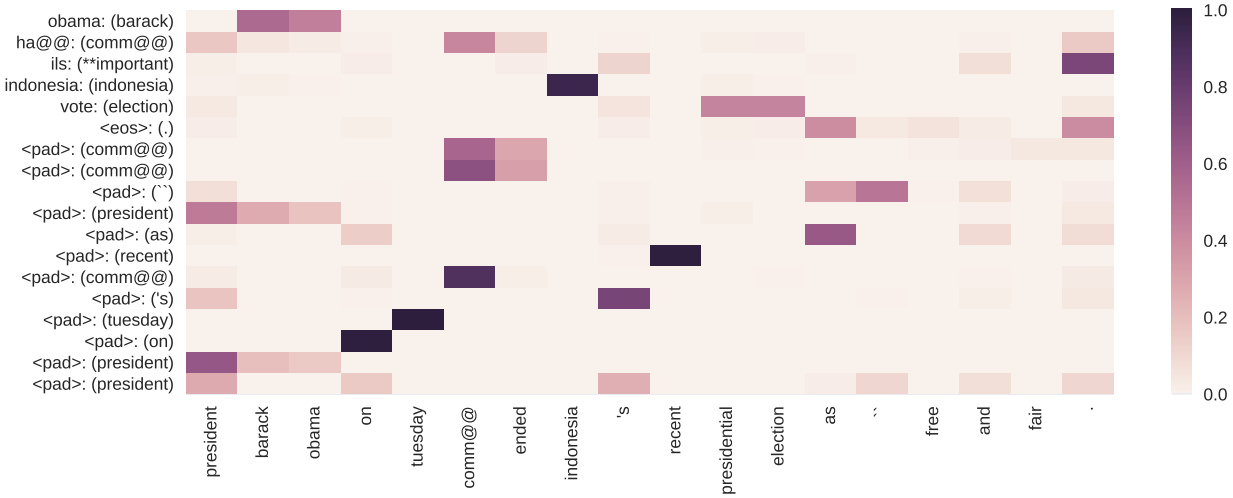
Figure 5: Although "london" is not at the beginning of the source sentence, the SPM aligns "london" in the source and the target. On the other hand, EncDec concentrates most of the attention at the end of the sentence. As a result, most of the target-side tokens are aligned with the sentence period of the source sentence.

Type	Aligned Pairs: (Target-side Token, SPM Prediction)
Verb Inflection	(calls, called), (release, released), (win, won), (condemns, condemned), (rejects, rejected), (warns, warned)
Paraphrasing to Shorter Form	(rules, agreement), (ends, closed), (keep, continued), (sell, issue), (quake, earthquake), (eu, european)
Others	(tourists, people), (dead, killed), (dead, died), (administration, bush), (aircraft, planes), (militants, group)

Table 4: Examples of the alignment that the SPM acquired

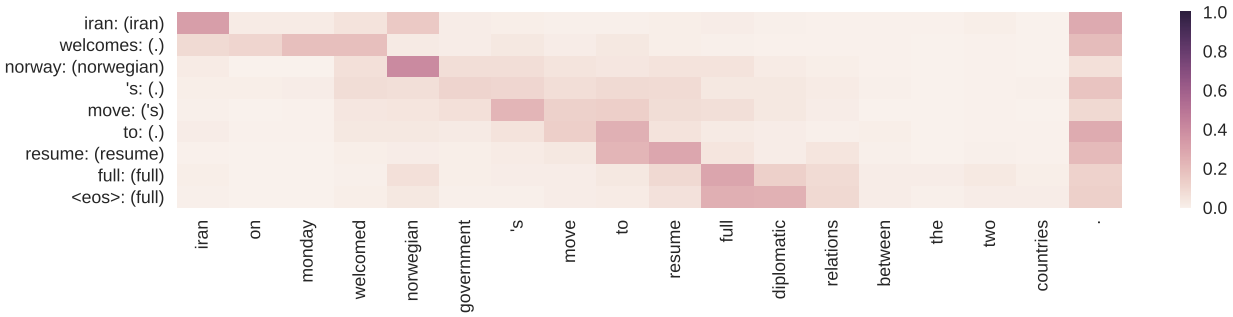


(a) Attention distribution of EncDec

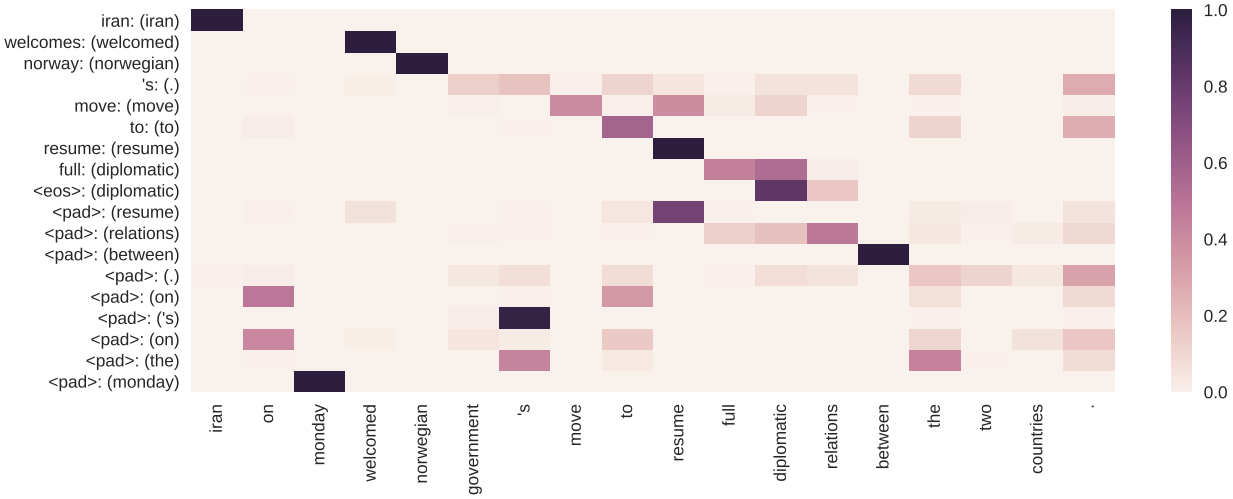


(b) Source-side prediction of EncDec+SPM

Figure 6: SPM aligns “election” with “vote”, whereas EncDec aligns “vote” with sentence period.



(a) Attention distribution of EncDec



(b) Source-side prediction of EncDec+SPM

Figure 7: The SPM aligns "welcomes" with "welcomed." On the other hand, EncDec aligns "welcomes" with the sentence period.