A Unified Model for Extractive and Abstractive Summarization using Inconsistency Loss

Wan-Ting Hsu[1], **Chieh-Kai Lin**[1], **Ming-Ying Lee**[1], **Kerui Min**[2], **Jing Tang**[2], **Min Sun**[1]
[1] National Tsing Hua University, [2] Cheetah Mobile
{hsuwanting, axk51013, masonyl03}@gapp.nthu.edu.tw,
{minkerui, tangjing}@cmcm.com, sunmin@ee.nthu.edu.tw

arXiv:1805.06266v1 [cs.CL] 16 May 2018

Abstract

We propose a unified model combining the strength of extractive and abstractive summarization. On the one hand, a simple extractive model can obtain sentence-level attention with high ROUGE scores but less readable. On the other hand, a more complicated abstractive model can obtain word-level dynamic attention to generate a more readable paragraph. In our model, sentence-level attention is used to modulate the word-level attention such that words in less attended sentences are less likely to be generated. Moreover, a novel inconsistency loss function is introduced to penalize the inconsistency between two levels of attentions. By end-to-end training our model with the inconsistency loss and original losses of extractive and abstractive models, we achieve state-of-the-art ROUGE scores while being the most informative and readable summarization on the CNN/Daily Mail dataset in a solid human evaluation.

1 Introduction

Text summarization is the task of automatically condensing a piece of text to a shorter version while maintaining the important points. The ability to condense text information can aid many applications such as creating news digests, presenting search results, and generating reports. There are mainly two types of approaches: extractive and abstractive. Extractive approaches assemble summaries directly from the source text typically selecting one whole sentence at a time. In contrast, abstractive approaches can generate novel words and phrases not copied from the source text.

Original Article: McDonald's says..... The company says it expects the new 'Artisan Grilled Chicken' to be in its more than 14,300 U.S. stores by the end of next week, in products including a new sandwich, as well as existing sandwiches, wraps and salads. It says the biggest change is the removal of sodium phosphates, which it said was used to keep the chicken moist, in favor of vegetable starch. The new recipe also does not use maltodextrin, which McDonald's said is generally used as a sugar to increase browning or as a carrier for seasoning. Jessica Foust, director of culinary innovation at McDonald's, said the changes were made because customers said they want 'simple, clean ingredients' they are familiar with..... And *Panera Bread has said* it plans to purge artificial colors, flavors and preservatives from its food by 2016.....

Extractive Approach: **The company** says it expects the new 'Artisan Grilled Chicken' to be in its more than 14,300 U.S. stores by the end of next week, in products including a new sandwich, as well as existing sandwiches, wraps and salads. It says the biggest change is the removal of sodium phosphates, **which it said was used to keep the chicken moist, in favor of vegetable starch**. The new recipe also does not use maltodextrin, which McDonald's said is generally used as a sugar to increase browning or as a carrier for seasoning.

Abstractive Approach: McDonald's says it expects the new 'Artisan Grilled Chicken' to be in its more than 14,300 U.S. stores by the end of next week. The company says the changes were made because customers said they want 'simple, clean ingredients' they are familiar with. *McDonald's said* it plans to purge artificial colors, flavors and preservatives from its food by 2016.

Unified Approach: **McDonald's says it expects** the new 'Artisan Grilled Chicken' to be in its more than 14,300 U.S. stores by the end of next week, in products including a new sandwich, as well as existing sandwiches, wraps and salads. It says the biggest change is the removal of sodium phosphates. The new recipe also does not use maltodextrin, which McDonald's said is generally used as a sugar to increase browning or as a carrier for seasoning.

Figure 1: Comparison of extractive, abstractive, and our unified summaries on a news article. The extractive model picks most important but **incoherent or not concise** (see blue bold font) sentences. The abstractive summary is readable, concise but still *loses or mistakes some facts* (see red italics font). The final summary rewritten from fragments (see underline font) has the advantages from both extractive (importance) and abstractive advantage (**coherence** (see green bold font)).

Hence, abstractive summaries can be more coherent and concise than extractive summaries.

Extractive approaches are typically simpler. They output the probability of each sentence to be selected into the summary. Many earlier works on summarization (Cheng and Lapata, 2016; Nallapati et al., 2016a, 2017; Narayan et al., 2017; Yasunaga et al., 2017) focus on extractive summarization. Among them, Nallapati et al.

(2017) have achieved high ROUGE scores. On the other hand, abstractive approaches (Nallapati et al., 2016b; See et al., 2017; Paulus et al., 2017; Fan et al., 2017; Liu et al., 2017) typically involve sophisticated mechanism in order to paraphrase, generate unseen words in the source text, or even incorporate external knowledge. Neural networks (Nallapati et al., 2017; See et al., 2017) based on the attentional encoder-decoder model (Bahdanau et al., 2014) were able to generate abstractive summaries with high ROUGE scores but suffer from inaccurately reproducing factual details and an inability to deal with out-of-vocabulary (OOV) words. Recently, See et al. (2017) propose a pointer-generator model which has the abilities to copy words from source text as well as generate unseen words. Despite recent progress in abstractive summarization, extractive approaches (Nallapati et al., 2017; Yasunaga et al., 2017) and lead-3 baseline (i.e., selecting the first 3 sentences) still achieve strong performance in ROUGE scores.

We propose to explicitly take advantage of the strength of state-of-the-art extractive and abstractive summarization and introduced the following unified model. Firstly, we treat the probability output of each sentence from the extractive model (Nallapati et al., 2017) as sentence-level attention. Then, we modulate the word-level dynamic attention from the abstractive model (See et al., 2017) with sentence-level attention such that words in less attended sentences are less likely to be generated. In this way, extractive summarization mostly benefits abstractive summarization by mitigating spurious word-level attention. Secondly, we introduce a novel inconsistency loss function to encourage the consistency between two levels of attentions. The loss function can be computed without additional human annotation and has shown to ensure our unified model to be mutually beneficial to both extractive and abstractive summarization. On CNN/Daily Mail dataset, our unified model achieves state-of-the-art ROUGE scores and outperforms a strong extractive baseline (i.e., lead-3). Finally, to ensure the quality of our unified model, we conduct a solid human evaluation and confirm that our method significantly outperforms recent state-of-the-art methods in informativity and readability.

To summarize, our contributions are twofold:

- We propose a unified model combining sentence-level and word-level attentions to take advantage of both extractive and abstractive summarization approaches.
- We propose a novel inconsistency loss function to ensure our unified model to be mutually beneficial to both extractive and abstractive summarization. The unified model with inconsistency loss achieves the best ROUGE scores on CNN/Daily Mail dataset and outperforms recent state-of-the-art methods in informativity and readability on human evaluation.

2 Related Work

Text summarization has been widely studied in recent years. We first introduce the related works of neural-network-based extractive and abstractive summarization. Finally, we introduce a few related works with hierarchical attention mechanism.

Extractive summarization. Kågebäck et al. (2014) and Yin and Pei (2015) use neural networks to map sentences into vectors and select sentences based on those vectors. Cheng and Lapata (2016), Nallapati et al. (2016a) and Nallapati et al. (2017) use recurrent neural networks to read the article and get the representations of the sentences and article to select sentences. Narayan et al. (2017) utilize side information (i.e., image captions and titles) to help the sentence classifier choose sentences. Yasunaga et al. (2017) combine recurrent neural networks with graph convolutional networks to compute the salience (or importance) of each sentence. While some extractive summarization methods obtain high ROUGE scores, they all suffer from low readability.

Abstractive summarization. Rush et al. (2015) first bring up the abstractive summarization task and use attention-based encoder to read the input text and generate the summary. Based on them, Miao and Blunsom (2016) use a variational auto-encoder and Nallapati et al. (2016b) use a more powerful sequence-to-sequence model. Besides, Nallapati et al. (2016b) create a new article-level summarization dataset called CNN/Daily Mail by adapting DeepMind question-answering dataset (Hermann et al., 2015). Ranzato et al. (2015) change the traditional training method to directly optimize evaluation metrics (e.g., BLEU and ROUGE). Gu et al. (2016), See et al. (2017) and Paulus et al. (2017) combine pointer networks

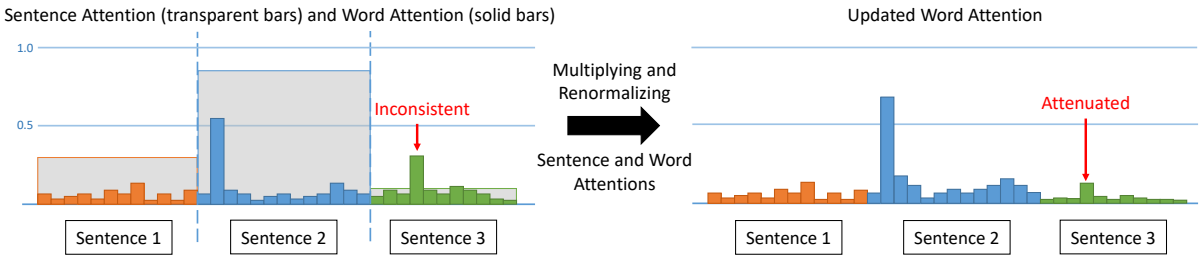

Figure 2: Our unified model combines the word-level and sentence-level attentions. Inconsistency occurs when word attention is high but sentence attention is low (see red arrow).

(Vinyals et al., 2015) into their models to deal with out-of-vocabulary (OOV) words. Chen et al. (2016) and See et al. (2017) restrain their models from attending to the same word to decrease repeated phrases in the generated summary. Paulus et al. (2017) use policy gradient on summarization and state out the fact that high ROUGE scores might still lead to low human evaluation scores. Fan et al. (2017) apply convolutional sequence-to-sequence model and design several new tasks for summarization. Liu et al. (2017) achieve high readability score on human evaluation using generative adversarial networks.

Hierarchical attention. Attention mechanism was first proposed by Bahdanau et al. (2014). Yang et al. (2016) proposed a hierarchical attention mechanism for document classification. We adopt the method of combining sentence-level and word-level attention in Nallapati et al. (2016b). However, their sentence attention is dynamic, which means it will be different for each generated word. Whereas our sentence attention is fixed for all generated words. Inspired by the high performance of extractive summarization, we propose to use fixed sentence attention.

Our model combines state-of-the-art extractive model (Nallapati et al., 2017) and abstractive model (See et al., 2017) by combining sentence-level attention from the former and word-level attention from the latter. Furthermore, we design an inconsistency loss to enhance the cooperation between the extractive and abstractive models.

3 Our Unified Model

We propose a unified model to combine the strength of both state-of-the-art extractor (Nallapati et al., 2017) and abstracter (See et al., 2017). Before going into details of our model, we first define the tasks of the extractor and abstracter.

Problem definition. The input of both extractor and abstracter is a sequence of words $\mathbf{w} = [w_1, w_2, ..., w_m, ...]$, where m is the word index. The sequence of words also forms a sequence of sentences $\mathbf{s} = [s_1, s_2, ..., s_n, ...]$, where n is the sentence index. The m^{th} word is mapped into the $n(m)^{th}$ sentence, where $n(\cdot)$ is the mapping function. The output of the extractor is the sentence-level attention $\boldsymbol{\beta} = [\beta_1, \beta_2, ..., \beta_n, ...]$, where β_n is the probability of the n^{th} sentence been extracted into the summary. On the other hand, our attention-based abstractor computes word-level attention $\boldsymbol{\alpha}^t = [\alpha_1^t, \alpha_2^t, ..., \alpha_m^t, ...]$ dynamically while generating the t^{th} word in the summary. The output of the abstracter is the summary text $\mathbf{y} = [y^1, y^2, ..., y^t, ...]$, where y^t is t^{th} word in the summary.

In the following, we introduce the mechanism to combine sentence-level and word-level attentions in Sec. 3.1. Next, we define the novel inconsistency loss that ensures extractor and abstracter to be mutually beneficial in Sec. 3.2. We also give the details of our extractor in Sec. 3.3 and our abstracter in Sec. 3.4. Finally, our training procedure is described in Sec. 3.5.

3.1 Combining Attentions

Pieces of evidence (e.g., Vaswani et al. (2017)) show that attention mechanism is very important for NLP tasks. Hence, we propose to explicitly combine the sentence-level β_n and word-level α_m^t attentions by simple scalar multiplication and renormalization. The updated word attention $\hat{\alpha}_m^t$ is

$$\hat{\alpha}_m^t = \frac{\alpha_m^t \times \beta_{n(m)}}{\sum_m \alpha_m^t \times \beta_{n(m)}}. \quad (1)$$

The multiplication ensures that only when both word-level α_m^t and sentence-level β_n attentions are high, the updated word attention $\hat{\alpha}_m^t$ can be high. Since the sentence-level attention β_n from the extractor already achieves high ROUGE

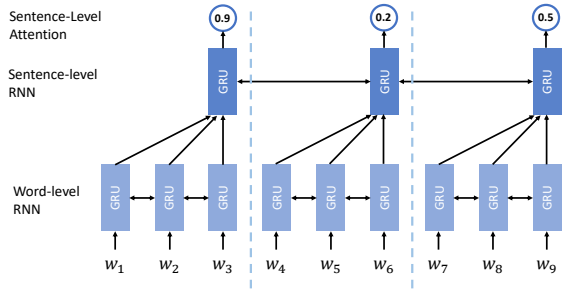

Figure 3: Architecture of the extractor. We treat the sigmoid output of each sentence as sentence-level attention $\in [0, 1]$.

scores, β_n intuitively modulates the word-level attention α_m^t to mitigate spurious word-level attention such that words in less attended sentences are less likely to be generated (see Fig. 2). As highlighted in Sec. 3.4, the word-level attention $\hat{\alpha}_m^t$ significantly affects the decoding process of the abstracter. Hence, an updated word-level attention is our key to improve abstractive summarization.

3.2 Inconsistency Loss

Instead of only leveraging the complementary nature between sentence-level and word-level attentions, we would like to encourage these two-levels of attentions to be mostly consistent to each other during training as an intrinsic learning target for free (i.e., without additional human annotation). Explicitly, we would like the sentence-level attention to be high when the word-level attention is high. Hence, we design the following inconsistency loss,

$$L_{inc} = -\frac{1}{T}\sum_{t=1}^{T}\log(\frac{1}{|\mathcal{K}|}\sum_{m\in\mathcal{K}}\alpha_m^t \times \beta_{n(m)}), \quad (2)$$

where $\mathcal{K}$ is the set of top K attended words and T is the number of words in the summary. This implicitly encourages the distribution of the word-level attentions to be sharp and sentence-level attention to be high. To avoid the degenerated solution for the distribution of word attention to be one-hot and sentence attention to be high, we include the original loss functions for training the extractor (L_{ext} in Sec. 3.3) and abstracter (L_{abs} and L_{cov} in Sec. 3.4). Note that Eq. 1 is the only part that the extractor is interacting with the abstracter. Our proposed inconsistency loss facilitates our end-to-end trained unified model to be mutually beneficial to both the extractor and abstracter.

3.3 Extractor

Our extractor is inspired by Nallapati et al. (2017). The main difference is that our extractor does not need to obtain the final summary. It mainly needs to obtain a short list of important sentences with a high recall to further facilitate the abstractor. We first introduce the network architecture and the loss function. Finally, we define our ground truth important sentences to encourage high recall.

Architecture. The model consists of a hierarchical bidirectional GRU which extracts sentence representations and a classification layer for predicting the sentence-level attention β_n for each sentence (see Fig. 3).

Extractor loss. The following sigmoid cross entropy loss is used,

$$L_{ext} = -\frac{1}{N}\sum_{n=1}^{N}(g_n \log\beta_n + (1-g_n)\log(1-\beta_n)), \quad (3)$$

where $g_n \in \{0, 1\}$ is the ground-truth label for the n^{th} sentence and N is the number of sentences. When $g_n = 1$, it indicates that the n^{th} sentence should be attended to facilitate abstractive summarization.

Ground-truth label. The goal of our extractor is to extract sentences with high informativity, which means the extracted sentences should contain information that is needed to generate an abstractive summary as much as possible. To obtain the ground-truth labels $\mathbf{g} = \{g_n\}_n$, first, we measure the informativity of each sentence s_n in the article by computing the ROUGE-L recall score (Lin, 2004) between the sentence s_n and the reference abstractive summary $\hat{\mathbf{y}} = \{\hat{y}^t\}_t$. Second, we sort the sentences by their informativity and select the sentence in the order of high to low informativity. We add one sentence at a time if the new sentence can increase the informativity of all the selected sentences. Finally, we obtain the ground-truth labels $\mathbf{g}$ and train our extractor by minimizing Eq. 3. Note that our method is different from Nallapati et al. (2017) who aim to extract a final summary for an article so they use ROUGE F-1 score to select ground-truth sentences; while we focus on high informativity, hence, we use ROUGE recall score to obtain as much information as possible with respect to the reference summary $\hat{\mathbf{y}}$.

3.4 Abstracter

The second part of our model is an abstracter that reads the article; then, generate a summary

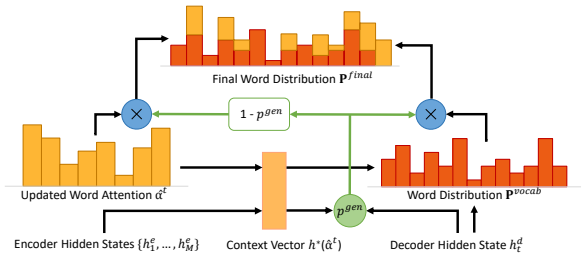

Figure 4: Decoding mechanism in the abstracter. In the decoder step t, our updated word attention $\hat{\boldsymbol{\alpha}}^t$ is used to generate context vector $h^*(\hat{\boldsymbol{\alpha}}^t)$. Hence, it updates the final word distribution $\mathbf{P}^{final}$.

word-by-word. We use the pointer-generator network proposed by See et al. (2017) and combine it with the extractor by combining sentence-level and word-level attentions (Sec. 3.1).

Pointer-generator network. The pointer-generator network (See et al., 2017) is a specially designed sequence-to-sequence attentional model that can generate the summary by copying words in the article or generating words from a fixed vocabulary at the same time. The model contains a bidirectional LSTM which serves as an encoder to encode the input words $\mathbf{w}$ and a unidirectional LSTM which serves as a decoder to generate the summary $\mathbf{y}$. For details of the network architecture, please refer to See et al. (2017). In the following, we describe how the updated word attention $\hat{\boldsymbol{\alpha}}^t$ affects the decoding process.

Notations. We first define some notations. h_m^e is the encoder hidden state for the m^{th} word. h_t^d is the decoder hidden state in step t. $h^*(\hat{\boldsymbol{\alpha}}^t) = \sum_m^M \hat{\alpha}_m^t \times h_m^e$ is the context vector which is a function of the updated word attention $\hat{\boldsymbol{\alpha}}^t$. $\mathbf{P}^{vocab}(h^*(\hat{\boldsymbol{\alpha}}^t))$ is the probability distribution over the fixed vocabulary before applying the copying mechanism.

$$\begin{aligned}&\mathbf{P}^{vocab}(h^*(\hat{\boldsymbol{\alpha}}^t)) \qquad (4)\\ &= \text{softmax}(W_2(W_1[h_t^d, h^*(\hat{\boldsymbol{\alpha}}^t)] + b_1) + b_2),\end{aligned}$$

where W_1, W_2, b_1 and b_2 are learnable parameters. $\mathbf{P}^{vocab} = \{P_w^{vocab}\}_w$ where $P_w^{vocab}(h^*(\hat{\boldsymbol{\alpha}}^t))$ is the probability of word w being decoded. $p^{gen}(h^*(\hat{\boldsymbol{\alpha}}^t)) \in [0, 1]$ is the generating probability (see Eq.8 in See et al. (2017)) and $1 - p^{gen}(h^*(\hat{\boldsymbol{\alpha}}^t))$ is the copying probability.

Final word distribution. $P_w^{final}(\hat{\boldsymbol{\alpha}}^t)$ is the final probability of word w being decoded (i.e., $y^t = w$). It is related to the updated word attention $\hat{\boldsymbol{\alpha}}^t$ as follows (see Fig. 4),

$$\begin{aligned}P_w^{final}(\hat{\boldsymbol{\alpha}}^t) &= p^{gen}(h^*(\hat{\boldsymbol{\alpha}}^t))P_w^{vocab}(h^*(\hat{\boldsymbol{\alpha}}^t)) \quad (5)\\ &+ (1 - p^{gen}(h^*(\hat{\boldsymbol{\alpha}}^t))) \sum_{m:w_m=w} \hat{\alpha}_m^t.\end{aligned}$$

Note that $\mathbf{P}^{final} = \{P_w^{final}\}_w$ is the probability distribution over the fixed vocabulary and out-of-vocabulary (OOV) words. Hence, OOV words can be decoded. Most importantly, it is clear from Eq. 5 that $P_w^{final}(\hat{\boldsymbol{\alpha}}^t)$ is a function of the updated word attention $\hat{\boldsymbol{\alpha}}^t$. Finally, we train the abstracter to minimize the negative log-likelihood:

$$L_{abs} = -\frac{1}{T}\sum_{t=1}^{T} \log P_{\hat{y}^t}^{final}(\hat{\boldsymbol{\alpha}}^t)\,, \qquad (6)$$

where $\hat{y}^t$ is the t^{th} token in the reference abstractive summary.

Coverage mechanism. We also apply coverage mechanism (See et al., 2017) to prevent the abstracter from repeatedly attending to the same place. In each decoder step t, we calculate the coverage vector $\mathbf{c}^t = \sum_{t'=1}^{t-1} \hat{\boldsymbol{\alpha}}^{t'}$ which indicates so far how much attention has been paid to every input word. The coverage vector $\mathbf{c}^t$ will be used to calculate word attention $\hat{\boldsymbol{\alpha}}^t$ (see Eq.11 in See et al. (2017)). Moreover, coverage loss L_{cov} is calculated to directly penalize the repetition in updated word attention $\hat{\boldsymbol{\alpha}}^t$:

$$L_{cov} = \frac{1}{T}\sum_{t=1}^{T}\sum_{m=1}^{M} \min(\hat{\alpha}_m^t, c_m^t)\,. \qquad (7)$$

The objective function for training the abstracter with coverage mechanism is the weighted sum of negative log-likelihood and coverage loss.

3.5 Training Procedure

We first pre-train the extractor by minimizing L_{ext} in Eq. 3 and the abstracter by minimizing L_{abs} and L_{cov} in Eq. 6 and Eq. 7, respectively. When pre-training, the abstracter takes ground-truth extracted sentences (i.e., sentences with $g_n = 1$) as input. To combine the extractor and abstracter, we proposed two training settings : (1) two-stages training and (2) end-to-end training.

Two-stages training. In this setting, we view the sentence-level attention $\boldsymbol{\beta}$ from the pre-trained extractor as hard attention. The extractor becomes a classifier to select sentences with high attention (i.e., $\beta_n >$ threshold). We simply combine the extractor and abstracter by feeding the extracted sentences to the abstracter. Note that we finetune the abstracter since the input text becomes extractive summary which is obtained from the extractor.

End-to-end training. For end-to-end training, the sentence-level attention $\boldsymbol{\beta}$ is soft attention and will be combined with the word-level attention $\boldsymbol{\alpha}^t$ as described in Sec. 3.1. We end-to-end train the extractor and abstracter by minimizing four loss functions: L_{ext}, L_{abs}, L_{cov}, as well as L_{inc} in Eq. 2. The final loss is as below:

$$L_{e2e} = \lambda_1 L_{ext} + \lambda_2 L_{abs} + \lambda_3 L_{cov} + \lambda_4 L_{inc}, \tag{8}$$

where λ_1, λ_2, λ_3, λ_4 are hyper-parameters. In our experiment, we give L_{ext} a bigger weight (e.g., $\lambda_1 = 5$) when end-to-end training with L_{inc} since we found that L_{inc} is relatively large such that the extractor tends to ignore L_{ext}.

4 Experiments

We introduce the dataset and implementation details of our method evaluated in our experiments.

4.1 Dataset

We evaluate our models on the CNN/Daily Mail dataset (Hermann et al., 2015; Nallapati et al., 2016b; See et al., 2017) which contains news stories in CNN and Daily Mail websites. Each article in this dataset is paired with one human-written multi-sentence summary. This dataset has two versions: *anonymized* and *non-anonymized*. The former contains the news stories with all the named entities replaced by special tokens (e.g., `@entity2`); while the latter contains the raw text of each news story. We follow See et al. (2017) and obtain the *non-anonymized* version of this dataset which has 287,113 training pairs, 13,368 validation pairs and 11,490 test pairs.

4.2 Implementation Details

We train our extractor and abstracter with 128-dimension word embeddings and set the vocabulary size to 50k for both source and target text. We follow Nallapati et al. (2017) and See et al. (2017) and set the hidden dimension to 200 and 256 for the extractor and abstracter, respectively. We use Adagrad optimizer (Duchi et al., 2011) and apply early stopping based on the validation set. In the testing phase, we limit the length of the summary to 120.

Pre-training. We use learning rate 0.15 when pre-training the extractor and abstracter. For the extractor, we limit both the maximum number of sentences per article and the maximum number of tokens per sentence to 50 and train the model for 27k iterations with the batch size of 64. For the abstracter, it takes ground-truth extracted sentences (i.e., sentences with $g_n = 1$) as input. We limit the length of the source text to 400 and the length of the summary to 100 and use the batch size of 16. We train the abstracter without coverage mechanism for 88k iterations and continue training for 1k iterations with coverage mechanism ($L_{abs} : L_{cov} = 1 : 1$).

Two-stages training. The abstracter takes extracted sentences with $\beta_n > 0.5$, where $\boldsymbol{\beta}$ is obtained from the pre-trained extractor, as input during two-stages training. We finetune the abstracter for 10k iterations.

End-to-end training. During end-to-end training, we will minimize four loss functions (Eq. 8) with $\lambda_1 = 5$ and $\lambda_2 = \lambda_3 = \lambda_4 = 1$. We set K to 3 for computing L_{inc}. Due to the limitation of the memory, we reduce the batch size to 8 and thus use a smaller learning rate 0.01 for stability. The abstracter here reads the whole article. Hence, we increase the maximum length of source text to 600. We end-to-end train the model for 50k iterations.

5 Results

Our unified model not only generates an abstractive summary but also extracts the important sentences in an article. Our goal is that both of the two types of outputs can help people to read and understand an article faster. Hence, in this section, we evaluate the results of our extractor in Sec. 5.1 and unified model in Sec. 5.2. Furthermore, in Sec. 5.3, we perform human evaluation and show that our model can provide a better abstractive summary than other baselines.

5.1 Results of Extracted Sentences

To evaluate whether our extractor obtains enough information for the abstracter, we use full-length ROUGE recall scores[1] between the extracted sentences and reference abstractive summary. High ROUGE recall scores can be obtained if the extracted sentences include more words or sequences overlapping with the reference abstractive summary. For each article, we select sentences with the sentence probabilities β greater than 0.5. We show the results of the ground-truth sentence labels (Sec. 3.3) and our models on the

[1] All our ROUGE scores are reported by the official ROUGE script. We use the `pyrouge` package. https://pypi.org/project/pyrouge/0.1.3/

Method	ROUGE-1	ROUGE-2	ROUGE-L
pre-trained	73.50	35.55	68.57
end2end w/o inconsistency loss	72.97	35.11	67.99
end2end w/ inconsistency loss	**78.40**	**39.45**	**73.83**
ground-truth labels	89.23	49.36	85.46

Table 1: ROUGE recall scores of the extracted sentences. *pre-trained* indicates the extractor trained on the ground-truth labels. *end2end* indicates the extractor after end-to-end training with the abstracter. Note that *ground-truth labels* show the upper-bound performance since the reference summary to calculate ROUGE-recall is abstractive. All our ROUGE scores have a 95% confidence interval with at most $\pm$0.33.

Method	ROUGE-1	ROUGE-2	ROUGE-L
HierAttn (Nallapati et al., 2016b)*	32.75	12.21	29.01
DeepRL (Paulus et al., 2017)*	39.87	15.82	36.90
pointer-generator (See et al., 2017)	39.53	17.28	36.38
GAN (Liu et al., 2017)	39.92	17.65	36.71
two-stage (ours)	39.97	17.43	36.34
end2end w/o inconsistency loss (ours)	40.19	17.67	36.68
end2end w/ inconsistency loss (ours)	**40.68**	**17.97**	**37.13**
lead-3 (See et al., 2017)	40.34	17.70	36.57

Table 2: ROUGE F-1 scores of the generated abstractive summaries on the CNN/Daily Mail test set. Our two-stages model outperforms pointer-generator model on ROUGE-1 and ROUGE-2. In addition, our model trained end-to-end with inconsistency loss exceeds the lead-3 baseline. All our ROUGE scores have a 95% confidence interval with at most $\pm$0.24. '*' indicates the model is trained and evaluated on the anonymized dataset and thus is not strictly comparable with ours.

test set of the CNN/Daily Mail dataset in Table 1. Note that the ground-truth extracted sentences can't get ROUGE recall scores of 100 because reference summary is abstractive and may contain some words and sequences that are not in the article. Our extractor performs the best when end-to-end trained with inconsistency loss.

5.2 Results of Abstractive Summarization

We use full-length ROUGE-1, ROUGE-2 and ROUGE-L F-1 scores to evaluate the generated summaries. We compare our models (two-stage and end-to-end) with state-of-the-art abstractive summarization models (Nallapati et al., 2016b; Paulus et al., 2017; See et al., 2017; Liu et al., 2017) and a strong lead-3 baseline which directly uses the first three article sentences as the summary. Due to the writing style of news articles, the most important information is often written at the beginning of an article which makes lead-3 a strong baseline. The results of ROUGE F-1 scores are shown in Table 2. We prove that with help of the extractor, our unified model can outperform pointer-generator (the third row in Table 2) even with two-stages training (the fifth row in Table 2). After end-to-end training without inconsistency loss, our method already achieves better ROUGE scores by cooperating with each other. Moreover, our model end-to-end trained with inconsistency loss achieves state-of-the-art ROUGE scores and exceeds lead-3 baseline.

In order to quantify the effect of inconsistency loss, we design a metric – inconsistency rate R_{inc} – to measure the inconsistency for each generated summary. For each decoder step t, if the word with maximum attention belongs to a sentence with low attention (i.e., $\beta_{n(\text{argmax}(\boldsymbol{\alpha}^t))} < \text{mean}(\boldsymbol{\beta})$), we define this step as an inconsistent step t_{inc}. The inconsistency rate R_{inc} is then defined as the percentage of the inconsistent steps in the summary.

$$R_{inc} = \frac{\text{Count}(t_{inc})}{T}, \tag{9}$$

where T is the length of the summary. The average inconsistency rates on test set are shown in Table 4. Our inconsistency loss significantly decrease R_{inc} from about 20% to 4%. An example of inconsistency improvement is shown in Fig. 5.

Method	informativity	conciseness	readability
DeepRL (Paulus et al., 2017)	3.23	2.97	2.85
pointer-generator (See et al., 2017)	3.18	3.36	3.47
GAN (Liu et al., 2017)	3.22	3.52	3.51
Ours	**3.58**	3.40	**3.70**
reference	3.43	**3.61**	3.62

Table 3: Comparing human evaluation results with state-of-the-art methods.

Method	avg. R_{inc}
w/o incon. loss	0.198
w/ incon. loss	0.042

Table 4: Inconsistency rate of our end-to-end trained model with and without inconsistency loss.

Without inconsistency loss:
If that was a tornado, it was one monster of one. Luckily, so far it looks like no one was hurt. With tornadoes touching down near Dallas on Sunday, **Ryan Shepard snapped a photo of a black cloud formation reaching down to the ground. He said it was a tornado. It wouldn't be an exaggeration to say it looked half a mile wide.** More like a mile, said Jamie Moore, head of emergency management in Johnson County, Texas. It could have been one the National Weather Service warned about in a tweet as severe thunderstorms drenched the area, causing street flooding. (...)

With inconsistency loss:
If that was a tornado, it was one monster of one. Luckily, so far it looks like no one was hurt. With **tornadoes touching down near Dallas on Sunday, Ryan Shepard snapped a photo of a black cloud formation reaching down to the ground.** He said it was a tornado. It wouldn't be an exaggeration to say it looked half a mile wide. More like a mile, said Jamie Moore, head of emergency management in Johnson County, Texas. **It could have been one the National Weather Service warned about in a tweet as** severe thunderstorms drenched the area, **causing street flooding.** (...)

Figure 5: Visualizing the consistency between sentence and word attentions on the original article. We highlight word (bold font) and sentence (underline font) attentions. We compare our methods trained with and without inconsistency loss. Inconsistent fragments (see red bold font) occur when trained without the inconsistency loss.

5.3 Human Evaluation

We perform human evaluation on Amazon Mechanical Turk (MTurk)[2] to evaluate the informativity, conciseness and readability of the summaries. We compare our best model (end2end with inconsistency loss) with pointer-generator (See et al., 2017), generative adversarial network (Liu et al., 2017) and deep reinforcement model (Paulus et al., 2017). For these three models, we use the test set outputs provided by the authors[3].

[2] https://www.mturk.com/

[3] https://github.com/abisee/pointer-generator and https://likicode.com for the first two. For DeepRL, we asked through email.

We randomly pick 100 examples in the test set. All generated summaries are re-capitalized and de-tokenized. Since Paulus et al. (2017) trained their model on anonymized data, we also recover the anonymized entities and numbers of their outputs.

We show the article and 6 summaries (reference summary, 4 generated summaries and a random summary) to each human evaluator. The random summary is a reference summary randomly picked from other articles and is used as a trap. We show the instructions of three different aspects as: (1) Informativity: how well does the summary capture the important parts of the article? (2) Conciseness: is the summary clear enough to explain everything without being redundant? (3) Readability: how well-written (fluent and grammatical) the summary is? The user interface of our human evaluation is shown in the supplementary material.

We ask the human evaluator to evaluate each summary by scoring the three aspects with 1 to 5 score (higher the better). We reject all the evaluations that score the informativity of the random summary as 3, 4 and 5. By using this trap mechanism, we can ensure a much better quality of our human evaluation. For each example, we first ask 5 human evaluators to evaluate. However, for those articles that are too long, which are always skipped by the evaluators, it is hard to collect 5 reliable evaluations. Hence, we collect at least 3 evaluations for every example. For each summary, we average the scores over different human evaluators.

The results are shown in Table 3. The reference summaries get the best score on conciseness since the recent abstractive models tend to copy sentences from the input articles. However, our model learns well to select important information and form complete sentences so we even get slightly better scores on informativity and readability than the reference summaries. We show a typical example of our model comparing with other state-of-

Original article (truncated):
A chameleon balances carefully on a branch, waiting calmly for its prey... except that if you look closely, you will see that this picture is not all that it seems. For the 'creature' poised to pounce is not a colourful species of lizard but something altogether more human. Featuring two carefully painted female models, it is a clever piece of sculpture designed to create an amazing illusion. It is the work of Italian artist Johannes Stoetter. Scroll down for video. Can you see us? Italian artist **Johannes Stoetter has painted two naked women to look like a chameleon. The 37-year-old has previously transformed his models into frogs and parrots** but this may be his most intricate and impressive piece to date. **Stoetter daubed water-based body paint on the naked models to create the multicoloured effect, then intertwined them to form the shape of a chameleon.** To complete the deception, the models rested on a bench painted to match their skin and held the green branch in the air beneath them. Stoetter can take weeks to plan one of his pieces and hours to paint it. Speaking about The Chameleon, he said: 'I worked about four days to design the motif bigger and paint it with colours. The body painting took me about six hours with the help of an assistant. I covered the hair with natural clay to make the heads look bald.' Camouflage job: A few finishing touches are applied to the two naked models to complete the transformation. 'There are different difficulties on different levels as in every work, but I think that my passion and love to my work is so big, that I figure out a way to deal with difficulties. My main inspirations are nature, my personal life-philosophy, every-day-life and people themselves.' However, the finished result existed only briefly before the models were able to get up and wash the paint off with just a video and some photographs to record it. (...)

Reference summary:
Johannes Stoetter's artwork features two carefully painted female models. The 37-year-old has previously transformed models into frogs and parrots. Daubed water-based body paint on naked models to create the effect. Completing the deception, models rested on bench painted to match skin.

DeepRL:
Italian artist Johannes Stoetter has painted female models to look like a chameleon. The 37-year-old has previously transformed his models into frogs and parrots but *this may be the work of Italian artist*. He has painted nude models and it is a clever piece of sculpture designed to create an amazing illusion. *It is work of artist Johannes Stoetter.*

GAN:
Italian artist Johannes Stoetter has painted two naked women to look like a chameleon. The 37-year-old has previously transformed his models into frogs and parrots *but this may be his most intricate and impressive piece to date*.

Pointer-generator:
Italian artist Johannes Stoetter has painted two naked women to look like a chameleon. *It is the work of Italian artist Johannes Stoetter.* Stoetter daubed water-based body paint on the naked models to create the multicoloured effect, then intertwined them to form the shape of a chameleon.

Our unified model (with inconsistency loss):
Johannes Stoetter has painted two naked women to look like a chameleon. The 37-year-old has previously transformed his models into frogs and parrots. Stoetter daubed water-based body paint on the naked models to create the multicoloured effect, then intertwined them to form the shape of a chameleon.

Figure 6: Typical Comparison. Our model attended at the most important information (blue bold font) matching well with the reference summary; while other state-of-the-art methods generate repeated or less important information (red italic font).

the-art methods in Fig. 6. More examples (5 using CNN/Daily Mail news articles and 3 using non-news articles as inputs) are provided in the supplementary material.

6 Conclusion

We propose a unified model combining the strength of extractive and abstractive summarization. Most importantly, a novel inconsistency loss function is introduced to penalize the inconsistency between two levels of attentions. The inconsistency loss enables extractive and abstractive summarization to be mutually beneficial. By end-to-end training of our model, we achieve the best ROUGE-recall and ROUGE while being the most informative and readable summarization on the CNN/Daily Mail dataset in a solid human evaluation.

Acknowledgments

We thank the support from Cheetah Mobile, National Taiwan University, and MOST 107-2634-F-007-007, 106-3114-E-007-004, 107-2633-E-002-001. We thank Yun-Zhu Song for assistance with useful survey and experiment on the task of abstractive summarization.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 2015 International Conference on Learning Representations*.

Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, and Hui Jiang. 2016. Distraction-based neural networks for modeling documents. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16)*.

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.

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.

Angela Fan, David Grangier, and Michael Auli. 2017. Controllable abstractive summarization. *arXiv preprint arXiv:1711.05217*.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1631–1640.

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*, pages 1693–1701.

Mikael Kågebäck, Olof Mogren, Nina Tahmasebi, and Devdatt Dubhashi. 2014. Extractive summarization using continuous vector space models. In *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)*, pages 31–39.

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

Linqing Liu, Yao Lu, Min Yang, Qiang Qu, Jia Zhu, and Hongyan Li. 2017. Generative adversarial network for abstractive text summarization. In *Proceddings of the 2018 Association for the Advancement of Artificial Intelligence*.

Yishu Miao and Phil Blunsom. 2016. Language as a latent variable: Discrete generative models for sentence compression. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 319–328.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Proceddings of the 2017 Association for the Advancement of Artificial Intelligence*, pages 3075–3081.

Ramesh Nallapati, Bowen Zhou, and Mingbo Ma. 2016a. Classify or select: Neural architectures for extractive document summarization. *arXiv preprint arXiv:1611.04244*.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Caglar Gulcehre, and Bing Xiang. 2016b. 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.

Shashi Narayan, Nikos Papasarantopoulos, Mirella Lapata, and Shay B Cohen. 2017. Neural extractive summarization with side information. *arXiv preprint arXiv:1704.04530*.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. In *Proceedings of the 2018 International Conference on Learning Representations*.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*.

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.

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)*, volume 1, pages 1073–1083.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489.

Michihiro Yasunaga, Rui Zhang, Kshitijh Meelu, Ayush Pareek, Krishnan Srinivasan, and Dragomir Radev. 2017. Graph-based neural multi-document summarization. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 452–462.

Wenpeng Yin and Yulong Pei. 2015. Optimizing sentence modeling and selection for document summarization. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, pages 1383–1389. AAAI Press.