Efficient and Interpretable Compressive Text Summarisation with Unsupervised Dual-Agent Reinforcement Learning

Peggy Tang[†], **Junbin Gao**[‡], **Lei Zhang**[◇], **Zhiyong Wang**[†]
[†]School of Computer Science, The University of Sydney
[◇]International Digital Economy Academy
[‡]The University of Sydney Business School, The University of Sydney
{peggy.tang, junbin.gao, zhiyong.wang}@sydney.edu.au,
leizhang@idea.edu.cn

Abstract

Recently, compressive text summarisation offers a balance between the conciseness issue of extractive summarisation and the factual hallucination issue of abstractive summarisation. However, most existing compressive summarisation methods are supervised, relying on the expensive effort of creating a new training dataset with corresponding compressive summaries. In this paper, we propose an efficient and interpretable compressive summarisation method that utilises unsupervised dual-agent reinforcement learning to optimise a summary's semantic coverage and fluency by simulating human judgment on summarisation quality. Our model consists of an extractor agent and a compressor agent, and both agents have a multi-head attentional pointer-based structure. The extractor agent first chooses salient sentences from a document, and then the compressor agent compresses these extracted sentences by selecting salient words to form a summary without using reference summaries to compute the summary reward. To our best knowledge, this is the first work on unsupervised compressive summarisation. Experimental results on three widely used datasets (e.g., Newsroom, CNN/DM, and XSum) show that our model achieves promising performance and a significant improvement on Newsroom in terms of the ROUGE metric, as well as interpretability of semantic coverage of summarisation results. [1]

1 Introduction

Most existing works on neural text summarisation are extractive, abstractive, and compressive-based. Extractive methods select salient sentences from a document to form its summary and ensure the production of grammatically and factually correct summaries. These methods usually follow the sentence ranking conceptualisation (Narayan et al., 2018b; Liu and Lapata, 2019; Zhong et al., 2020). The supervised models commonly rely on creating proxy extractive training labels for training (Nallapati et al., 2017; Jia et al., 2021; Mao et al., 2022; Klaus et al., 2022), which can be noisy and may not be reliant. Various unsupervised methods (Zheng and Lapata, 2019; Xu et al., 2020; Padmakumar and He, 2021; Liu et al., 2021) were proposed to leverage pre-trained language models to compute sentences similarities and select important sentences. Although these methods have significantly improved summarisation performance, the redundant information that appears in the salient sentences may not be minimized effectively.

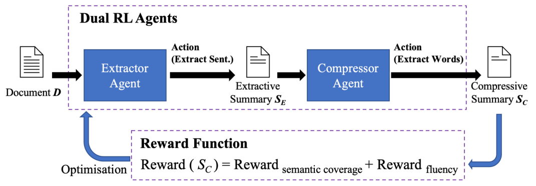


Figure 1: Illustration of our proposed URLComSum.

Abstractive methods formulate the task as a sequence-to-sequence generation task, with the document as the input sequence and the summary as the output sequence (See et al., 2017; Zhang et al., 2020; Wang et al., 2021; Liu et al., 2022) As supervised learning with ground-truth summaries may not provide useful insights on human judgment approximation, reinforcement training was proposed to optimise the ROUGE metric (Parnell et al., 2021), and to fine-tune a pre-trained language model (Laban et al., 2020). Prior studies showed that these generative models are highly prone to external hallucination (Maynez et al., 2020).

Compressive summarisation is a recent appraoch which aims to select words, instead of sentences, from an input document to form a summary, which improves the factuality and conciseness of a summary. The formulation of compressive document summarisation is usually a two-stage extract-then-

[1]Our source code is publicly available for research purposes at https://github.com/peggypytang/URLComSum/

compress approach (Zhang et al., 2018; Mendes et al., 2019; Xu and Durrett, 2019; Desai et al., 2020): it first extracts salient sentences from a document, then compresses the extracted sentences to form its summary. Most of these methods are supervised, which require a parallel dataset with document-summary pairs to train. However, the ground-truth summaries of existing datasets are usually abstractive-based and do not contain supervision information needed for extractive summarisation or compressive summarisation (Xu and Durrett, 2019; Mendes et al., 2019; Desai et al., 2020).

Therefore, to address these limitations, we propose a novel unsupervised compressive summarisation method with dual-agent reinforcement learning strategy to mimic human judgment, namely URLComSum. As illustrated in Figure 1, URLComSum consists of two modules, an extractor agent and a compressor agent. We model the sentence and word representations using a efficient Bi-LSTM (Graves and Schmidhuber, 2005) with multi-head attention (Vaswani et al., 2017) to capture both the long-range dependencies and the relationship between each word and each sentence. We use a pointer network (Vinyals et al., 2015) to find the optimal subset of sentences and words to be extracted since the Pointer Network is well-known for tackling combinatorial optimization problems. The extractor agent uses a hierarchical multi-head attentional Bi-LSTM model for learning the sentence representation of the input document and a pointer network for extracting the salient sentences of a document given a length budget. To further compress these extracted sentences all together, the compressor agent uses a multi-head attentional Bi-LSTM model for learning the word representation and a pointer network for selecting the words to assemble a summary.

As an unsupervised method, URLComSum does not require a parallel training dataset.We propose an unsupervised reinforcement learning training procedure to mimic human judgment: to reward the model that achieves high summary quality in terms of semantic coverage and language fluency. Inspired by Word Mover's Distance (Kusner et al., 2015), the semantic coverage rewardis measured by Wasserstein distance (Peyré et al., 2019) between the semantic distribution of the document and that of the summary. The fluency reward is measured by Syntactic Log-Odds Ratio (SLOR) (Pauls and Klein, 2012). SLOR is a referenceless fluency evaluation metric, which is effective in sentence compression (Kann et al., 2018) and has better correlation to human acceptability judgments (Lau et al., 2017).

The key contributions of this paper are:

- We propose the first unsupervised compressive summarisation method with dual-agent reinforcement learning, namely URLComSum.
- We design an efficient and interpretable multi-head attentional pointer-based neural network for learning the representation and for extracting salient sentences and words.
- We propose to mimic human judgment by optimising summary quality in terms of the semantic coverage reward, measured by Wasserstein distance, and the fluency reward, measured by Syntactic Log-Odds Ratio (SLOR).
- Comprehensive experimental results on three widely used datasets, including CNN/DM, XSum, Newsroom, demonstrate that URLComSum achieves great performance.

2 Related Work

Most of the existing works on neural text summarisation are extractive, abstractive, and compressive-based.

2.1 Extractive Methods

Extractive methods usually follow the sentence ranking conceptualisation, and an encoder-decoder scheme is generally adopted. An encoder formulates document or sentence representations, and a decoder predicts extraction classification labels. The supervised models commonly rely on creating proxy extractive training labels for training (Cheng and Lapata, 2016; Nallapati et al., 2017; Jia et al., 2021), which can be noisy and may not be reliant. Some methods were proposed to tackle this issue by training with reinforcement learning (Narayan et al., 2018b; Luo et al., 2019) to optimise the ROUGE metric directly. Various unsupervised methods (Zheng and Lapata, 2019; Xu et al., 2020; Padmakumar and He, 2021) were also proposed to leverage pre-trained language models to compute sentences similarities and select important sentences. Although these methods have significantly improved summarisation performance, since the entire sentences are extracted individually, the redundant information that appears in the salient sentences may not be minimized effectively.

2.2 Abstractive Methods

Abstractive methods formulate text summarisation as a sequence-to-sequence generation task, with the source document as the input sequence and the summary as the output sequence. Most existing methods follow the supervised RNN-based encoder-decoder framework (See et al., 2017; Zhang et al., 2020; Wang et al., 2021; Liu et al., 2022). As supervised learning with ground-truth summaries may not provide useful insights on human judgment approximation, reinforcement training was proposed to optimise the ROUGE metric (Paulus et al., 2018; Parnell et al., 2021), and to fine-tune a pre-trained language model (Laban et al., 2020). These models naturally learn to integrate knowledge from the training data while generating an abstractive summary. Prior studies showed that these generative models are highly prone to external hallucination, thus may generate contents that are unfaithful to the original document (Maynez et al., 2020).

2.3 Compressive Methods

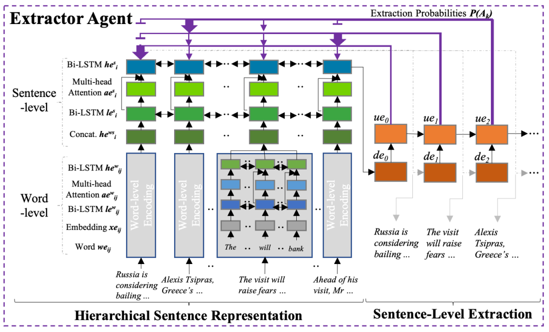
Compressive methods select words from a given document to assemble a summary. Due to the lack of training dataset, not until recently there have emerged works for compressive summarisation (Zhang et al., 2018; Mendes et al., 2019; Xu and Durrett, 2019; Desai et al., 2020). The formulation of compressive document summarisation is usually a two-stage extract-then-compress approach: it first extracts salient sentences from a document, then compresses the extracted sentences to form its summary. Most of these methods are supervised, which require a parallel dataset with document-summary pairs to train. However, the ground-truth summaries of existing datasets are usually abstractive-based and do not contain supervision information needed for extractive summarisation or compressive summarisation. Several reinforcement learning based methods (Zhang et al., 2018) use existing abstractive-based datasets for training, which is not aligned for compression. Note that existing compressors often perform compression sentence by sentence. As a result, the duplicated information among multiple sentences could be overlooked. Therefore, to address these limitations, we propose a novel unsupervised compressive method by exploring the dual-agent reinforcement learning strategy to mimic human judgment and perform text compression instead of sentence compression.

3 Methodology

As shown in Figure 1, our proposed compressive summarisation method, namely URLComSum, consists of two components, an extractor agent and a compressor agent. Specifically, the extractor agent selects salient sentences from a document $\mathbf{D}$ to form an extractive summary $\mathbf{S_E}$, and then the compressor agent compresses $\mathbf{S_E}$ by selecting words to assemble a compressive summary $\mathbf{S_C}$.

3.1 Extractor Agent

Given a document $\mathbf{D}$ consisting of a sequence of M sentences $\{\mathbf{s}_i | i = 1, ..., M\}$, and each sentence $\mathbf{s}_i$ consisting of a sequence of N words $\{\mathbf{we}_{ij} | j = 1, ..., N\}$[2], the extractor agent aims to produce an extractive summary $\mathbf{S_E}$ by learning sentence representation and selecting L_E sentences from $\mathbf{D}$. As illustrated in Figure 2, we design a hierarchical multi-head attentional sequential model for learning the sentence representations of the document and using a Pointer Network to extract sentences based on their representations.



Extractor Agent
Extraction Probabilities $P(A_k)$
Sentence -level
Bi-LSTM he^s_i
Multi-head Attention ae^s_i
Bi-LSTM le^s_i
Concat. he^{ws}_i
Word -level
Bi-LSTM he^w_{ij}
Multi-head Attention ae^w_{ij}
Bi-LSTM le^w_{ij}
Embedding xe_{ij}
Word we_{ij}
Word-level Encoding
The
will
bank
Russia is considering bailing ...
Alexis Tsipras, Greece's ...
The visit will raise fears ...
Ahead of his visit, Mr ...
ue_0
ue_1
ue_2
de_0
de_1
de_2
Russia is considering bailing ...
The visit will raise fears ...
Alexis Tsipras, Greece's ...
Hierarchical Sentence Representation
Sentence-Level Extraction

Figure 2: Illustration of the extractor agent.

3.1.1 Hierarchical Sentence Representation

To model the local context of each sentence and the global context between sentences, we use two-levels Bi-LSTMs to model this hierarchical structure, one at the word level to encode the word sequence of each sentence, one at the sentence level to encode the sentence sequence of the document. To model the context-dependency of the importance of words and sentences, we apply two levels of multi-head attention mechanism (Vaswani et al., 2017), one at each of the two-level Bi-LSTMs.

[2]We have pre-fixed the length of each sentence and each document by padding.

Given a sentence $\mathbf{s}_i$, we encode its words into word embeddings $\mathbf{xe}_i = \{\mathbf{xe}_{ij} | j = 1, ..., N\}$ by $\mathbf{xe}_{ij} = Enc(\mathbf{we}_{ij})$, where $Enc()$ denotes a word embedding lookup table. Then the sequence of word embeddings are fed into the word-level Bi-LSTM to produce an output representation of the words $\mathbf{le}^w$:

$$\mathbf{le}_{ij}^w = \overleftrightarrow{\mathrm{LSTM}}(\mathbf{xe}_{ij}), j \in [1, N] \,. \tag{1}$$

To utilize the multi-head attention mechanism to obtain $\mathbf{ae}_i^w = \{\mathbf{ae}_{i1}^w, ..., \mathbf{ae}_{iN}^w\}$ at word level, we define $Q_i = \mathbf{le}_i^w$, $K_i = V_i = \mathbf{xe}_i$,

$$\mathbf{ae}_i^w = \mathrm{MultiHead}(Q_i, K_i, V_i) \,. \tag{2}$$

The concatenation of $\mathbf{le}_i^w$ and $\mathbf{ae}_i^w$ of the words are fed into a Bi-LSTM and the output is concatenated to obtain the local context representation $\mathbf{he}_i^{ws}$ for each sentence $\mathbf{s_i}$:

$$\begin{aligned} \mathbf{he}_{ij}^w &= \overleftrightarrow{\mathrm{LSTM}}([\mathbf{le}_{ij}^w; \mathbf{ae}_{ij}^w]), j \in [1, N] \,, \\ \mathbf{he}_i^{ws} &= [\mathbf{he}_{i1}^w, ..., \mathbf{he}_{iN}^w] \,. \end{aligned} \tag{3}$$

To further model the global context between sentences, we apply a similar structure at sentence level. $\mathbf{he}^{ws} = \{\mathbf{he}_i^{ws} | i = 1, ..., M\}$ are fed into the sentence-level Bi-LSTM to produce output representation of the sentences $\mathbf{le}^s$:

$$\mathbf{le}_i^s = \overleftrightarrow{\mathrm{LSTM}}(\mathbf{he}_i^{ws}), i \in [1, M] \,. \tag{4}$$

To utilize the multi-head attention mechanism to obtain $\mathbf{ae}^s = \{\mathbf{ae}_1^s, ..., \mathbf{ae}_M^s\}$ at sentence level, we define $Q = \mathbf{le}^s$, $K = V = \mathbf{he}^{ws}$,

$$\mathbf{ae}^s = \mathrm{MultiHead}(Q, K, V). \tag{5}$$

The concatenation of the Bi-LSTM output $\mathbf{le}^s$ and the multi-head attention output $\mathbf{ae}^s$ of the sentences are fed into a Bi-LSTM to obtain the final representations of sentences $\mathbf{he}^s = \{\mathbf{he}_1^s, ..., \mathbf{he}_M^s\}$:

$$\mathbf{he}_i^s = \overleftrightarrow{\mathrm{LSTM}}([\mathbf{le}_i^s; \mathbf{ae}_i^s]), i \in [1, M] \,. \tag{6}$$

3.1.2 Sentence-Level Extraction

Similar to (Chen and Bansal, 2018), we use an LSTM-based Pointer Network to decode the above sentence representations $\mathbf{he}^s = \{\mathbf{he}_1^s, ..., \mathbf{he}_M^s\}$ and extract sentences recurrently to form an extractive summary $\mathbf{S_E} = \{A_1, ..., A_k, ..., A_{L_E}\}$ with L_E sentences, where A_k denotes the k-th sentence extracted.

At the k-th time step, the pointer network receives the sentence representation of the previous extracted sentence and has hidden state de_k. It first obtains a context vector de_k' by attending to $\mathbf{he}^s$:

$$\begin{aligned} \mathbf{ue}_i^k &= v^T \tanh(W_1 \mathbf{he}_i^s + W_2 de_k), i \in (1, ..., M) \,, \\ \mathbf{ae}_i^k &= \mathrm{softmax}(\mathbf{ue}_i^k), i \in (1, ..., M) \,, \\ de_k' &= \sum_{i=1}^{M} \mathbf{ae}_i^k \mathbf{he}_i^s \,, \end{aligned} \tag{7}$$

where v, W_1, W_2 are learnable parameters of the pointer network. Then it predicts the extraction probability $p(A_k)$ of a sentence:

$$\begin{aligned} & de_k \leftarrow [de_k, de_k'] \,, \\ & \mathbf{ue}_i^k = v^T \tanh(W_1 \mathbf{he}_i^s + W_2 de_k), i \in (1, ..., M) \,, \\ & p(A_k | A_1, ..., A_{k-1}) = \mathrm{softmax}(\mathbf{ue}^k) \,. \end{aligned} \tag{8}$$

Decoding iterates until L_E sentences are selected to form S_E.

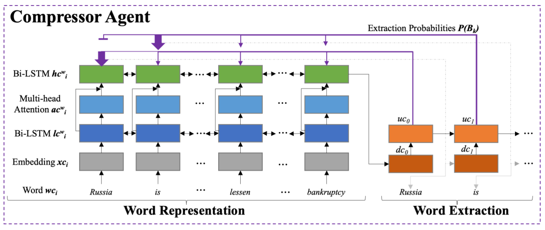


Figure 3: Illustration of the compressor agent.

3.2 Compressor Agent

Given an extractive summary $\mathbf{S_E}$ consisting of a sequence of words $\mathbf{wc} = \{\mathbf{wc}_i | i = 1, ..., N\}$, the compressor agent aims to produce a compressive summary $\mathbf{S_C}$ by selecting L_C words from $\mathbf{S_E}$. As illustrated in Figure 3, it has a multi-head attentional Bi-LSTM model to learn the word representations. It uses a pointer network to extract words based on their representations.

3.2.1 Word Representation

Given a sequence of words $\mathbf{wc}$, we encode the words into word embeddings $\mathbf{xc} = \{\mathbf{xc}_i | i = 1, ..., N\}$ by $\mathbf{xc}_i = Enc(\mathbf{wc}_i)$. Then the sequence of word embeddings are fed into a Bi-LSTM to produce the words' output representation $\mathbf{lc}^w$:

$$\mathbf{lc}_i^w = \overleftrightarrow{\mathrm{LSTM}}(\mathbf{xc}_i), i \in [1, N] \,. \tag{9}$$

To utilise the multi-head attention mechanism to obtain $\mathbf{ac}^w = \{\mathbf{ac}_1^w, ..., \mathbf{ac}_N^w\}$, we define $Q = \mathbf{lc}^w$, $K = V = \mathbf{xc}$,

$$\mathbf{ac}^w = \mathrm{MultiHead}(Q, K, V). \tag{10}$$

The concatenation of $\mathbf{lc}^w$ and $\mathbf{ac}^w$ of the words are fed into a Bi-LSTM to obtain the representation $\mathbf{hc}_i^w$ for each word $\mathbf{wc_i}$:

$$\mathbf{hc}_i^w = \overleftrightarrow{\text{LSTM}}([\mathbf{lc}_i^w; \mathbf{ac}_i^w]), i \in [1, N] . \tag{11}$$

3.2.2 Word-Level Extraction

The word extractor of the compressor agent shares the same structure as that of the extractor agent's sentence extractor. To select the words based on the above word representations $\mathbf{hc}^w = \{\mathbf{hc}_1^w, ..., \mathbf{hc}_N^w\}$, the word extractor decodes and extracts words recurrently to produce $\{B_1, ..., B_k, ..., B_{L_C}\}$, where B_k denotes the word extracted at the k-th time step. The selected words are reordered by their locations in the input document and assembled to form the compressive summary $\mathbf{S_C}$.

3.3 Reward in Reinforcement Learning

We use the compressive summary $\mathbf{S_C}$ to compute the reward of reinforcement learning and denote Reward$(\mathbf{D}, \mathbf{S_C})$ as Reward$(\mathbf{D}, \mathbf{S})$ for simplicity. Reward$(\mathbf{D}, \mathbf{S})$ is a weighted sum of the semantic coverage award Reward$_{\text{cov}}(\mathbf{D}, \mathbf{S})$ and the fluency reward Reward$_{\text{flu}}(\mathbf{S})$:

$$\begin{aligned}\text{Reward}(\mathbf{D}, \mathbf{S}) = w_{\text{cov}} &\text{Reward}_{\text{cov}}(\mathbf{D}, \mathbf{S}) \\ &+ w_{\text{flu}} \text{Reward}_{\text{flu}}(\mathbf{S}) ,\end{aligned} \tag{12}$$

where w_{cov} and w_{flu} denote the weights of two rewards.

3.3.1 Semantic Coverage Reward

We compute Reward$_{\text{cov}}$ with the Wasserstein distance between the corresponding semantic distributions of the document $\mathbf{D}$ and the summary $\mathbf{S}$, which is the minimum cost required to transport the semantics from $\mathbf{D}$ to $\mathbf{S}$. We denote $\mathbf{D} = \{d_i | i = 1, ..., N\}$ to represent a document, where d_i indicates the count of the i-th token (i.e., word or phrase in a vocabulary of size N). Similarly, for a summary $\mathbf{S} = \{s_j | j = 1, ..., N\}$, s_j is respect to the count of the j-th token . The semantic distribution of a document is characterized in terms of normalised term frequency without the stop-words. The term frequency of the i-th token in the document $\mathbf{D}$ and the j-th token in the summary $\mathbf{S}$ are denoted as TF$_\mathbf{D}(i)$ and TF$_\mathbf{S}(j)$, respectively. By defining TF$_\mathbf{D} = \{\text{TF}_\mathbf{D}(i)\} \in \mathbf{R}^N$ and TF$_\mathbf{S} = \{\text{TF}_\mathbf{S}(j)\} \in \mathbf{R}^N$, we have the semantic distributions within $\mathbf{D}$ and $\mathbf{S}$ respectively.

The transportation cost matrix $\mathbf{C}$ is obtained by measuring the semantic similarity between each of the tokens. Given a pre-trained tokeniser and token embedding model with N tokens, define $\mathbf{v}_i$ to represent the feature embedding of the i-th token. Then the transport cost c_{ij} from the i-th to the j-th token is computed based on the cosine similarity: $c_{ij} = 1 - \frac{<\mathbf{v}_i, \mathbf{v}_j>}{\|\mathbf{v}_i\|_2 \|\mathbf{v}_j\|_2}$. An optimal transport plan $\mathbf{T}^* = \{t_{i,j}^*\} \in \mathbf{R}^{N \times N}$ in pursuit of minimizing the transportation cost can be obtained by solving the optimal transportation and resources allocation optimization problem (Peyré et al., 2019). Note that the transport plan can be used to interpret the transportation of tokens from document to summary, which brings interpretability to our URLComSum method.

Wasserstein distance measuring the distance between the two semantic distributions TF$_\mathbf{D}$ and TF$_\mathbf{S}$ with the optimal transport plan is computed by: $d_W(\text{TF}_\mathbf{D}, \text{TF}_\mathbf{S} | \mathbf{C}) = \sum_{i,j} t_{ij}^* c_{ij}$. Reward$_{\text{cov}}(\mathbf{D}, \mathbf{S})$ can be further defined as:

$$\text{Reward}_{\text{cov}}(\mathbf{D}, \mathbf{S}) = 1 - d_W(\text{TF}_\mathbf{D}, \text{TF}_\mathbf{S} | \mathbf{C}) . \tag{13}$$

3.3.2 Fluency Reward

We utilise Syntactic Log-Odds Ratio (SLOR) (Pauls and Klein, 2012) to measure Reward$_{\text{flu}}(S)$, which is defined as: Reward$_{\text{flu}}(S) = \frac{1}{|S|}(\log(P_{LM}(S)) - \log(P_U(S)))$, where $P_{LM}(S)$ denotes the probability of the summary assigned by a pre-trained language model LM, $p_U(S) = \prod_{t \in S} P(t)$ denotes the unigram probability for rare word adjustment, and $|S|$ denotes the sentence length.

We use the Self-Critical Sequence Training (SCST) method (Rennie et al., 2017), since this training algorithm has demonstrated promising results in text summarisation (Paulus et al., 2018; Laban et al., 2020). For a given input document, the model produces two separate output summaries: the sampled summary $\mathbf{S}^s$, obtained by sampling the next pointer t_i from the probability distribution at each time step i, and the baseline summary $\hat{\mathbf{S}}$, obtained by always picking the most likely next pointer t at each i. The training objective is to minimise the following loss:

$$\begin{aligned}Loss = -(\text{Reward}(\mathbf{D}, \mathbf{S}^s) - \text{Reward}(\mathbf{D}, \hat{\mathbf{S}})) \\ \cdot \frac{1}{N} \sum_{i=1}^{N} \log p(t_i^s | t_1^s, ..., t_{i-1}^s, \mathbf{D}) ,\end{aligned} \tag{14}$$

where N denotes the length of the pointer sequence, which is the number of extracted sentences for the extractor agent and the number of extracted words for the compressor agent.

Minimising the loss is equivalent to maximising the conditional likelihood of $\mathbf{S}^s$ if the sampled summary $\mathbf{S}^s$ outperforms the baseline summary $\hat{\mathbf{S}}$, i.e. $\text{Reward}(\mathbf{D}, \mathbf{S}^s) - \text{Reward}(\mathbf{D}, \hat{\mathbf{S}}) > 0$, thus increasing the expected reward of the model.

4 Experiments

4.1 Experimental Settings

We conducted comprehensive experiments on three widely used datasets: *Newsroom* (Grusky et al., 2018), *CNN/DailyMail (CNN/DM)* (Hermann et al., 2015), and *XSum* (Narayan et al., 2018a). We set the LSTM hidden size to 150 and the number of recurrent layers to 3. We performed hyperparameter searching for w_{cov} and w_{flu} and decided to set $w_{\text{cov}} = 1$, $w_{\text{flu}} = 2$ in all our experiments since it provides more balanced results across the datasets. We trained the URLComSum with AdamW (Loshchilov and Hutter, 2018) with learning rate 0.01 with a batch size of 3. We obtained the word embedding from the pre-trained GloVe (Pennington et al., 2014). We used BERT for the pre-trained embedding models used for computing semantic coverage reward. We chose GPT2 for the trained language model used for computing the fluency reward due to strong representation capacity.

As shown in Table 1, we followed (Mendes et al., 2019) to set $\mathbf{L_E}$ for Newsroom and (Zhong et al., 2020) to set $\mathbf{L_E}$ for CNN/DM and XSum. We also followed their protocols to set $\mathbf{L_C}$ by matching the average number of words in summaries.

Dataset	**Newsroom**	**CNN/DM**	**XSum**
#Sentences in Doc.	27	39	19
#Tokens in Doc.	659	766	367
$\mathbf{L_E}$	2	3	2
$\mathbf{L_C}$	26	58	24
Train	995,041	287,113	204,045
Test	108,862	11,490	11,334

Table 1: Overview of the three datasets. #Sentences in Doc. and #Tokens in Doc. denote the average number of sentences and words in the documents respectively. $\mathbf{L_E}$ denotes the number of sentences to be selected by the extractor agent. $\mathbf{L_C}$ denotes the number of words to be selected by the compressor agent. Train and Test denote the size of train and test sets.

Method	ROUGE-1	ROUGE-2	ROUGE-L
LEAD	33.9	23.2	30.7
LEAD-WORD	34.9	23.1	30.7
Supervised Methods			
EXCONSUMM (Ext.)*	31.9	16.3	26.9
EXCONSUMM (Ext.+Com.)*	25.5	11.0	21.1
Unsupervised Methods			
SumLoop (Abs.)	27.0	9.6	26.4
TextRank (Ext.)	24.5	10.1	20.1
URLComSum (Ext.)	33.9	**23.2**	30.0
URLComSum (Ext.+Com.)	**34.6**	22.9	**30.5**

Table 2: Comparisons on the **Newsroom** test set. The symbol * indicates that the model is not directly comparable to ours as it is based on a subset (the "Mixed") of the dataset.

Method	ROUGE-1	ROUGE-2	ROUGE-L
LEAD	40.0	17.5	32.9
LEAD-WORD	39.7	16.6	32.5
Supervised Methods			
LATENTCOM (Ext.)	41.1	18.8	37.5
LATENTCOM (Ext.+Com.)	36.7	15.4	34.3
JECS (Ext.)	40.7	18.0	36.8
JECS (Ext.+Com.)	41.7	18.5	37.9
EXCONSUMM (Ext.)	41.7	18.6	37.8
EXCONSUMM (Ext.+Com.)	40.9	18.0	37.4
CUPS (Ext.)	43.7	20.6	40.0
CUPS (Ext.+Com.)	44.0	20.6	40.4
Unsupervised Methods			
SumLoop (Abs.)	37.7	14.8	**34.7**
TextRank (Ext.)	34.1	12.8	22.5
PacSum (Ext.)	**40.3**	**17.6**	24.9
PMI (Ext.)	36.7	14.5	23.3
URLComSum (Ext.)	40.0	17.5	32.9
URLComSum (Ext.+Com.)	39.3	16.0	32.2

Table 3: Comparisons between our URLComSum and the state-of-the-art methods on the **CNN/DM** test set. (Ext.), (Abs.), and (Com.) denote the method is extractive, abstractive, and compressive respectively.

We compare our model with existing compressive methods which are all supervised, including *LATENTCOM* (Zhang et al., 2018), *EXCONSUMM* (Mendes et al., 2019), *JECS* (Xu and Durrett, 2019), *CUPS* (Desai et al., 2020). Since our method is unsupervised, we also compare it with unsupervised extractive and abstractive methods, including *TextRank* (Mihalcea and Tarau, 2004), *PacSum* (Zheng and Lapata, 2019), *PMI* (Padmakumar and He, 2021), and *SumLoop* (Laban et al., 2020). To better evaluate compressive methods, we followed a similar concept as LEAD baseline (See et al., 2017) and created *LEAD-WORD* baseline which

Method	ROUGE-1	ROUGE-2	ROUGE-L
LEAD	19.4	2.4	12.9
LEAD-WORD	18.3	1.9	12.8
Supervised Methods			
CUPS (Ext.)	24.2	5.0	18.3
CUPS (Ext.+Com.)	26.0	5.4	19.9
Unsupervised Methods			
TextRank (Ext.)	19.0	3.1	12.6
PacSum (Ext.)	**19.4**	2.7	12.4
PMI (Ext.)	19.1	**3.2**	12.5
URLComSum (Ext.)	**19.4**	2.4	**12.9**
URLComSum (Ext.+Com.)	18.0	1.8	12.7

Table 4: Comparisons on the **XSum** test set.URLComSum (Ext.) denotes the extractive summary produced by our extractor agent. URLComSum (Ext.+Com.) denotes the compressive summary produced further by our compressor agent.

extracts the first several words of a document as a summary. The commonly used ROUGE metric (Lin, 2004) is adopted.

4.2 Experimental Results

The experimental results of URLComSum on different datasets are shown in Table 2, Table 3 and Table 4 in terms of ROUGE-1, ROUGE-2 and ROUGE-L F-scores. (Ext.), (Abs.), and (Com.) denote that the method is extractive, abstractive, and compressive, respectively. Note that on the three datasets, LEAD and LEAD-WORD baseline are considered strong baselines in the literature and sometimes perform better than the state-of-the-art supervised and unsupervised models. As also discussed in (See et al., 2017; Padmakumar and He, 2021), it could be due to the Inverted Pyramid writing structure (Pöttker, 2003) of news articles, in which important information is often located at the beginning of an article and a paragraph.

Our URLComSum method significantly outperforms all the unsupervised and supervised ones on Newsroom. This demonstrates the effectiveness of our proposed method. Note that, unlike supervised EXCONSUMM, our reward strategy contributes to performance improvement when the compressor agent is utilised. For example, in terms of ROUGE-L, EXCONSUMM(Ext.+Com.) does not outperform EXCONSUMM(Ext.), while URLComSum(Ext.+Com.) outperforms URLComSum(Ext.). Similarly, our URLComSum method achieves the best performance among all the unsupervised methods on XSum, in terms of ROUGE-1 and -L. URLComSum underperforms in ROUGE-2, which may be due to the trade-off between informativeness and fluency. The improvement on Newsroom is greater than those on CNN/DM and XSum, which could be because the larger size of Newsroom is more helpful for training our model.

Our URLComSum method achieves comparable performance with other unsupervised methods on CNN/DM. Note that URLComSum does not explicitly take position information into account while some extractive methods take advantage of the lead bias of CNN/DM, such as PacSum and LEAD. Nevertheless, we observe that URLComSum(Ext.) achieves the same result as LEAD . Even though URLComSum is unsupervised, eventually the extractor agent learns to select the first few sentences of the documents, which follows the principle of the aforementioned Inverted Pyramid writing structure.

4.2.1 Ablation Studies

Effect of Compression. We observed that the extractive and compressive methods usually obtain better results than the abstractive ones in terms of ROUGE scores on CNN/DM and Newsroom, and vice versa on XSum. It may be that CNN/DM and Newsroom contain summaries that are usually more extractive, whereas XSum's summaries are highly abstractive. We noticed that URLComSum(Ext.+Com.) generally achieves higher ROUGE-1 and -L scores than its extractive version on Newsroom. Meanwhile, on CNN/DM and XSum, the compressive version has slightly lower ROUGE scores than the extractive version. We observe similar behaviour in the literature of compressive summarisation, which may be that the sentences of news articles have dense information and compression does not help much to further condense the content.

Effect of Transformer. Note that we investigated the popular transformer model (Vaswani et al., 2017) in our proposed framework to replace Bi-LSTM for learning the sentence and word representations. However, we noticed the transformer-based agents do not perform as well as the Bi-LSTM-based ones while training from scratch with the same training procedure. The difficulties of training a transformer model have also been discussed in (Popel and Bojar, 2018; Liu et al., 2020). Besides, the commonly used pre-trained transformer models, such as BERT (Devlin et al., 2019) and BART (Lewis et al., 2020), require high computational resources and usually use subword-based tokenizers. They are not suitable for URLComSum since our compressor agent points to words instead of subwords. Therefore, at this stage Bi-LSTM is a simpler and more efficient choice. Nevertheless, the transformer is a module that can be included in our framework and is worth further investigation in the future.

Comparison of Extraction, Abstraction and Compression Approaches. We observed that the extraction and compressive approaches usually obtain better results than the abstractive in terms of ROUGE scores on CNN/DM and Newsroom, and vice versa on XSum. It may be because CNN/DM and Newsroom contain summaries that are usually more extractive, whereas XSum's summaries are highly abstractive. Since the ROUGE metric reflects lexical matching only and overlooks the linguistic quality and factuality of the summary, it is difficult to conclude the superiority of one approach over the others solely based on the ROUGE scores. Automatic linguistic quality and factuality metrics would be essential to provide further insights and more meaningful comparisons.

4.3 Qualitative Analysis

In Figure 5, 6, 7 in Appendix A, summaries produced by URLComSum are shown together with the reference summaries of the sample documents in the CNN/DM, XSum, and Newsroom datasets. This demonstrates that our proposed URLComSum method is able to identify salient sentences and words and produce reasonably fluent summaries even without supervision information.

4.4 Interpretable Visualisation of Semantic Coverage

URLComSum is able to provide an interpretable visualisation of the semantic coverage on the summarisation results through the transportation matrix. Figure 4 illustrates the transport plan heatmap, which associated with a resulting summary is illustrated. A heatmap indicates the transportation of semantic contents between tokens in the document and its resulting summary. The higher the intensity, the more the semantic content of a particular document token is covered by a summary token. Red line highlights the transportation from the document to the summary of semantic content of token "country", which appears in both the document and the summary. Purple line highlights how the semantic content of token "debt", which appears in the document only but not the summary, are transported to token "bankruptcy" and "loans", which are semantically closer and have lower transport cost, and thus achieve a minimum transportation cost in the OT plan.

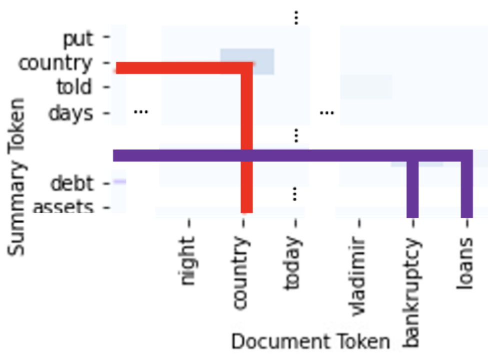


Figure 4: Interpretable visualisation of the OT plan. from a source document to a resulting summary on the CNN/DM dataset. The higher the intensity, the more the semantic content of a particular document token is covered by a summary token. Red line highlights the transportation from the document to the summary of semantic content of token "country", which appears in both the document and the summary. Purple line highlights how the semantic content of token "debt", which appears in the document only but not the summary, are transported to token "bankruptcy" and "loans", which are semantically closer and have lower transport cost, and thus achieve a minimum transportation cost in the OT plan.

5 Conclusion

In this paper, we have presented URLComSum, the first unsupervised and an efficient method for compressive text summarisation. Our model consists of dual agents: an extractor agent and a compressor agent. The extractor agent first chooses salient sentences from a document, and the compressor agent further select salient words from these extracted sentences to form a summary. To achieve unsupervised training of the extractor and compressor agents, we devise a reinforcement learning strategy to simulate human judgement on summary quality and optimize the summary's semantic coverage and fluency reward. Comprehensive experiments on three widely used benchmark datasets demonstrate the effectiveness of our proposed URLComSum and the great potential of unsupervised compressive summarisation. Our method provides interpretability of semantic coverage of summarisation results.

References

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 675–686, Melbourne, Australia. Association for Computational Linguistics.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Shrey Desai, Jiacheng Xu, and Greg Durrett. 2020. Compressive summarization with plausibility and salience modeling. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6259–6274, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks*, 18(5):602–610.

Max Grusky, Mor Naaman, and Yoav Artzi. 2018. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 708–719, New Orleans, Louisiana. Association for Computational Linguistics.

Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *International Conference on Neural Information Processing Systems (NeurIPS)*, page 1693–1701, Cambridge, MA, USA. MIT Press.

Ruipeng Jia, Yanan Cao, Haichao Shi, Fang Fang, Pengfei Yin, and Shi Wang. 2021. Flexible non-autoregressive extractive summarization with threshold: How to extract a non-fixed number of summary sentences. In *AAAI Conference on Artificial Intelligence*, volume 35, pages 13134–13142, Online. AAAI Press.

Katharina Kann, Sascha Rothe, and Katja Filippova. 2018. Sentence-level fluency evaluation: References help, but can be spared! In *Conference on Computational Natural Language Learning (CoNLL)*, pages 313–323, Brussels, Belgium. Association for Computational Linguistics.

Svea Klaus, Ria Van Hecke, Kaweh Djafari Naini, Ismail Sengor Altingovde, Juan Bernabé-Moreno, and Enrique Herrera-Viedma. 2022. Summarizing legal regulatory documents using transformers. In *International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, page 2426–2430, New York, NY, USA. Association for Computing Machinery.

Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. In *International Conference on Machine Learning (ICML)*, page 957–966, Lille, France. JMLR.org.

Philippe Laban, Andrew Hsi, John Canny, and Marti A. Hearst. 2020. The summary loop: Learning to write abstractive summaries without examples. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 5135–5150, Online. Association for Computational Linguistics.

Jey Han Lau, Alexander Clark, and Shalom Lappin. 2017. Grammaticality, acceptability, and probability: A probabilistic view of linguistic knowledge. *Cognitive Science*, 41(5):1202–1241.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Jingzhou Liu, Dominic J. D. Hughes, and Yiming Yang. 2021. Unsupervised extractive text summarization with distance-augmented sentence graphs. In *International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, page 2313–2317, New York, NY, USA. Association for Computing Machinery.

Liyuan Liu, Xiaodong Liu, Jianfeng Gao, Weizhu Chen, and Jiawei Han. 2020. Understanding the difficulty of training transformers. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

Yixin Liu, Pengfei Liu, Dragomir Radev, and Graham Neubig. 2022. BRIO: Bringing order to abstractive summarization. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 2890–2903, Dublin, Ireland. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, New Orleans, LA, USA. OpenReview.net.

Ling Luo, Xiang Ao, Yan Song, Feiyang Pan, Min Yang, and Qing He. 2019. Reading like HER: Human reading inspired extractive summarization. In *Conference on Empirical Methods in Natural Language Processing and International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Qianren Mao, Hongdong Zhu, Junnan Liu, Cheng Ji, Hao Peng, Jianxin Li, Lihong Wang, and Zheng Wang. 2022. Muchsum: Multi-channel graph neural network for extractive summarization. In *International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, page 2617–2622, New York, NY, USA. Association for Computing Machinery.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1906–1919, Online. Association for Computational Linguistics.

Afonso Mendes, Shashi Narayan, Sebastião Miranda, Zita Marinho, André F. T. Martins, and Shay B. Cohen. 2019. Jointly extracting and compressing documents with summary state representations. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 3955–3966, Minneapolis, Minnesota. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *AAAI Conference on Artificial Intelligence*, page 3075–3081, San Francisco, California, USA. AAAI Press.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018a. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018b. Ranking sentences for extractive summarization with reinforcement learning. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 1747–1759, New Orleans, Louisiana. Association for Computational Linguistics.

Vishakh Padmakumar and He He. 2021. Unsupervised extractive summarization using pointwise mutual information. In *Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 2505–2512, Online. Association for Computational Linguistics.

Jacob Parnell, Inigo Jauregi Unanue, and Massimo Piccardi. 2021. RewardsOfSum: Exploring reinforcement learning rewards for summarisation. In *Workshop on Structured Prediction for NLP (SPNLP)*, pages 1–11, Online. Association for Computational Linguistics.

Adam Pauls and Dan Klein. 2012. Large-scale syntactic language modeling with treelets. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 959–968, Jeju Island, Korea. Association for Computational Linguistics.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *International Conference on Learning Representations (ICLR)*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Gabriel Peyré, Marco Cuturi, et al. 2019. Computational optimal transport: With applications to data science. *Foundations and Trends in Machine Learning*, 11(5-6):355–607.

Martin Popel and Ondřej Bojar. 2018. Training tips for the transformer model. *The Prague Bulletin of Mathematical Linguistics (NeurIPS)*.

Horst Pöttker. 2003. News and its communicative quality: the inverted pyramid—when and why did it appear? *Journalism Studies*, 4(4):501–511.

Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

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 *International Conference on Neural Information Processing Systems (NeurIPS)*, volume 30, Long Beach, CA, USA. Curran Associates, Inc.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *International Conference on Neural Information Processing Systems (NeurIPS)*, volume 28, Montreal, Canada. Curran Associates, Inc.

Lihan Wang, Min Yang, Chengming Li, Ying Shen, and Ruifeng Xu. 2021. Abstractive text summarization with hierarchical multi-scale abstraction modeling and dynamic memory. In *International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, page 2086–2090, New York, NY, USA. Association for Computing Machinery.

Jiacheng Xu and Greg Durrett. 2019. Neural extractive text summarization with syntactic compression. In *Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3292–3303, Hong Kong, China. Association for Computational Linguistics.

Shusheng Xu, Xingxing Zhang, Yi Wu, Furu Wei, and Ming Zhou. 2020. Unsupervised extractive summarization by pre-training hierarchical transformers. In *Findings of the Association for Computational Linguistics: EMNLP*, pages 1784–1795, Online. Association for Computational Linguistics.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning (ICML)*, pages 11328–11339, Online. PMLR.

Xingxing Zhang, Mirella Lapata, Furu Wei, and Ming Zhou. 2018. Neural latent extractive document summarization. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 779–784, Brussels, Belgium. Association for Computational Linguistics.

Hao Zheng and Mirella Lapata. 2019. Sentence centrality revisited for unsupervised summarization. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 6236–6247, Florence, Italy. Association for Computational Linguistics.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2020. Extractive summarization as text matching. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 6197–6208, Online. Association for Computational Linguistics.

A Sample Summaries

The following shows the sample summaries generated by URLComSum on the CNN/DM, XSum, and Newsroom datasets. Sentences extracted by the URLComSum extractor agent are highlighted. Words selected by the URLComSum compressor agent are underlined in red. Our unsupervised method URLComSum can identify salient sentences and words to produce a summary with reasonable semantic coverage and fluency.

Source Document: Russia is considering bailing out Greece in exchange for the country's 'assets', it was reported last night. Alexis Tsipras, Greece's prime minister, will meet Vladimir Putin in Moscow today, amid reports that the Kremlin will offer controversial loans and discounts on supplies of natural gas in a bid to lessen its dependence on the West . The visit will raise fears the radical left government is looking east in search of alternative sources of finance as it bids to avoid bankruptcy. Scroll down for video . Alexis Tsipras, Greece's (...)
Reference Summary: Alexis Tsipras, Greece's prime minister, will meet Vladimir Putin in Moscow . The meeting comes amid reports Russia is considering bailing out Greece . Reports Kremlin may offer loans and discounts on supplies of natural gas .
URLComSum: Russia is considering bailing out Greece in exchange for the country ' s ' assets ' , it was reported last night . Alexis Tsipras , Greece ' s prime minister , will meet Vladimir Putin in Moscow today , amid reports that the Kremlin will offer controversial loans and discounts on supplies of natural gas in a bid its raise alternative as bids to avoid bankruptcy .

Figure 5: A sample summary produced by URLComSum on the CNN/DM dataset. The summary generated by URLComSum has ROUGE-1, ROUGE-2, and ROUGE-L F-Scores of 68.8, 52.7, and 62.4 respectively, with semantic coverage reward 0.76 and fluency reward 0.64, while the reference summary has semantic coverage reward 0.80 and fluency reward 0.62.

Source Document: Paul Robson is the second trader at the Dutch bank to plead guilty to trying to rig the Yen Libor rate and the first Briton to do so. Last year Rabobank paid $1bn (Â£597m) to US and European regulators for its part in the global rate-rigging scandal. Barclays Bank, Royal Bank of Scotland and Lloyds Bank have all previously been fined for rate rigging. (...)
Reference Summary: A former senior trader at Rabobank has pleaded guilty to interest rate rigging in the US.
URLComSum: Paul Robson is the second trader at the Dutch bank to plead guilty to trying to rig the Yen Libor rate and global rate-rigging scandal .

Figure 6: A sample summary produced by URLComSum on the XSum dataset. The summary generated by URLComSum has ROUGE-1, ROUGE-2, and ROUGE-L F-Scores of 38.1, 20.0, and 33.3 respectively, with semantic coverage reward 0.77 and fluency reward 0.56, while the reference summary has semantic coverage reward 0.73 and fluency reward 0.59.

Source Document: A man armed with a rifle has killed four people in a rampage in Girona province, north-east Spain, police say. The gunman walked into a bar in the town of Olot, 120km (70 miles) north of Barcelona, and shot two men - reportedly a father and son who were both construction workers. Minutes later, he went to a bank and killed two staff, police said. (...)
Reference Summary: A man armed with a rifle kills four people in a shooting rampage in north-east Spain, police say.
URLComSum: A man armed with a rifle has killed four people in a rampage in Girona province , north-east Spain , police say . The gunman walked into a bar in

Figure 7: A sample summary produced by URLComSum on the Newsroom dataset. The summary generated by URLComSum has ROUGE-1, ROUGE-2, and ROUGE-L F-Scores of 76.6, 62.2, and 76.6 respectively, with semantic coverage reward 0.79 and fluency reward 0.61, while the reference summary has semantic coverage reward 0.76 and fluency reward 0.65.