Discrete Optimization for Unsupervised Sentence Summarization with Word-Level Extraction

Raphael Schumann[1], **Lili Mou**[2], **Yao Lu**[3], **Olga Vechtomova**[3], **Katja Markert**[1]
[1]Institute of Computational Linguistics, Heidelberg University, Germany
{rschuman, markert}@cl.uni-heidelberg.de
[2]University of Alberta, Canada; Alberta Machine Intelligence Institute (Amii)
doublepower.mou@gmail.com
[3]University of Waterloo, Canada
{yao.lu, ovechtom}@uwaterloo.ca

Abstract

Automatic sentence summarization produces a shorter version of a sentence, while preserving its most important information. A good summary is characterized by language fluency and high information overlap with the source sentence. We model these two aspects in an unsupervised objective function, consisting of language modeling and semantic similarity metrics. We search for a high-scoring summary by discrete optimization. Our proposed method achieves a new state-of-the art for unsupervised sentence summarization according to ROUGE scores. Additionally, we demonstrate that the commonly reported ROUGE F1 metric is sensitive to summary length. Since this is unwillingly exploited in recent work, we emphasize that future evaluation should explicitly group summarization systems by output length brackets.[1]

1 Introduction

Sentence summarization transforms a long source sentence into a short summary, while preserving key information (Rush et al., 2015). Sentence summarization has wide applications, for example, news headline generation and text simplification.

State-of-the-art sentence summarization systems are based on sequence-to-sequence neural networks (Rush et al., 2015; Nallapati et al., 2016; Wang et al., 2019), which require massive parallel data for training. Therefore, unsupervised sentence summarization has recently attracted increasing interest. Cycle-consistency approaches treat the summary as a discrete latent variable and use it to reconstruct the source sentence (Wang and Lee, 2018; Baziotis et al., 2019). Such latent-space generation fails to explicitly model the resemblance between the source sentence and the target summary.

[1]Our code and system outputs are available at: https://github.com/raphael-sch/HC_Sentence_Summarization

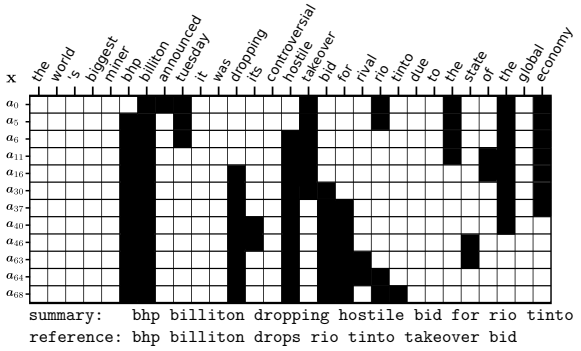

Figure 1: Summarizing a sentence $\mathbf{x}$ by hill climbing. Each row is a Boolean vector $\boldsymbol{a}_t$ at a search step t . A black cell indicates a word is selected, and *vice versa*. Randomly swapping two values in the Boolean vector yields a new summary that is scored by an objective function that measures language fluency and semantic similarity. If the new summary increases the objective, this summary is accepted as the current best solution. Rejected solutions are not depicted.

Zhou and Rush (2019) propose a left-to-right beam search approach based on a heuristically defined scoring function. However, beam search is biased towards the first few words of the source.

In this paper, we propose a hill-climbing approach to unsupervised sentence summarization, directly extracting words from the source sentence. This is motivated by the observation that human-written reference summaries exhibit high word overlap with the source sentence, even preserving word order to a large extent. To perform word extraction for summarization, we define a scoring function — similar to Miao et al. (2019) and Zhou and Rush (2019) — that evaluates the quality of a candidate summary by language fluency, semantic similarity to the source, and a hard constraint on output length. We search towards our scoring function by first choice hill-climbing (FCHC), shown in Figure 1. We start from a random subset of words of the required output length. For each search step, a new candidate is sampled by randomly swapping

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 5032–5042
July 5 - 10, 2020.

a selected word and a non-selected word. We accept the new candidate if its score is higher than the current one. In contrast to beam search (Zhou and Rush, 2019), our summary is not generated sequentially from the beginning of a sentence, and therefore not biased towards the first few words.

Due to the nature of the search action, our approach is able to *explicitly* control the length of a summary as a hard constraint. In all previous work, the summary length is weakly controlled by length embeddings or a soft length penalty (Zhou and Rush, 2019; Wang and Lee, 2018; Fevry and Phang, 2018; Baziotis et al., 2019). Thus, the generated summaries by different systems vary considerably in average length, for example, ranging from 9 to 15 on a headline corpus (Section 4.1). Previous work uses ROUGE F1 to compare summaries that might differ in length. We show that ROUGE F1 is unfortunately sensitive to summary output length, in general favoring models that produce longer summaries. Therefore, we argue that controlling the output length should be an integral part of the summarization task and that a fair system comparison can only be conducted between summaries in the same length bracket.

Our model establishes a new state-of-the-art for unsupervised sentence summarization across all commonly-used length brackets and different ROUGE metrics on the Gigaword dataset for headline generation (Rush et al., 2015) and on DUC2004 (Over and Yen, 2004).

The main contributions of this paper are:

- We propose a novel method for unsupervised sentence summarization by hill climbing with word-level extraction.
- We outperform current unsupervised sentence summarization systems, including more complex sentence reconstruction models.
- We show that ROUGE F1 is sensitive to summary length and thus emphasize the importance of explicitly controlling summary length for a fair comparison among different summarization systems.

2 Related Work

Text Summarization. The task can be categorized by source text types, such as *multi-document summarization* (Erkan and Radev, 2004; Radev et al., 2000; Haghighi and Vanderwende, 2009) and *single-document summarization* (Mihalcea and Tarau, 2004; Zhou and Hovy, 2004; Zheng and Lapata, 2019). Traditional approaches are mostly *extractive*, i.e., they extract entire sentences from a document. Recently, sequence-to-sequence (Seq2Seq) models have been used for *abstractive* summaries, where the system is able to synthesize new sentences (Nallapati et al., 2016, 2017; Gehrmann et al., 2018; Lewis et al., 2019; Fabbri et al., 2019). The copy mechanism (Gu et al., 2016) in a Seq2Seq model can be viewed as word-level extraction in abstractive summarization (See et al., 2017; Paulus et al., 2018). Both state-of-the-art extractive and abstractive approaches are usually supervised.

Sentence summarization yields a short summary for a long sentence. Hori and Furui (2004) and Clarke and Lapata (2006) extract single words from the source sentence based on language model fluency and linguistic constraints. They search via dynamic programming with a trigram language model, which restricts the model capacity. The Hedge Trimmer method (Dorr et al., 2003) also uses hand-crafted linguistic rules to remove constituents from a parse tree until a certain length is reached.

Rush et al. (2015) propose a supervised abstractive sentence summarization system with an attention mechanism (Bahdanau et al., 2015), and they also introduce a dataset for headline generation derived from Gigaword.[2] Subsequent models for this dataset were also supervised and mostly based on Seq2seq architectures (Nallapati et al., 2016; Chopra et al., 2016; Wang et al., 2019).

Recently, unsupervised approaches for sentence summarization have attracted increasing attention. Fevry and Phang (2018) learn a denoising autoencoder and control the summary length by a length embedding. Wang and Lee (2018) and Baziotis et al. (2019) use cycle-consistency (He et al., 2016) to learn the reconstruction of the source sentence and return the intermediate discrete representation as a summary. Zhou and Rush (2019) use beam search to optimize a scoring function, which considers language fluency and contextual matching.

Our work can be categorized under unsupervised sentence summarization. We accomplish this by word-level extraction from the source sentence.

Constrained Sentence Generation. Neural sentence generation is usually accomplished in an autoregressive way, for example, by recurrent neu-

[2] https://catalog.ldc.upenn.edu/LDC2003T05

ral networks generating words left-to-right. This is often enhanced by beam search (Sutskever et al., 2014), which keeps a beam of candidates in a partially greedy fashion. A few studies allow hard constraints on this decoding procedure. Hokamp and Liu (2017) use grid-beam search to impose lexical constraints during decoding. Anderson et al. (2017) propose constrained beam search to predict fixed image tags in an image transcription task. Miao et al. (2019) propose a Metropolis–Hastings sampler for sentence generation, where hard constraints can be incorporated into the target distribution. This is further extended to simulated annealing (Liu et al., 2020), or applied to the text simplification task (Kumar et al., 2020). Different from the above concurrent work, this paper applies the stochastic search framework to text summarization, and design our specific search space and search actions for word extraction.

In previous work on text summarization, length embeddings (Kikuchi et al., 2016; Fan et al., 2018) have been used to indicate the desired summary length. However, these are not hard constraints, because the model may learn to ignore such information.

3 Proposed Model

Given a source sentence $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ as input, our goal is to generate a shorter sentence $\mathbf{y} = (y_1, y_2, \ldots, y_m)$ as a summary of $\mathbf{x}$. We perform word-level extraction, in addition keeping the original word order intact. Thus, $\mathbf{y}$ is a subsequence of $\mathbf{x}$. Our word-level extraction optimizes a manually defined objective function $f(\mathbf{y}; \mathbf{x}, s)$, where the summary length s is predefined ($s < n$) and not subject to optimization. In the remainder of this section, we will describe the objective function, search space, and the search algorithm in detail.

3.1 Search Objective

We define an objective function $f(\mathbf{y}; \mathbf{x}, s)$, which our algorithm maximizes. It evaluates the fitness of a candidate sentence $\mathbf{y}$ as the summary of an input $\mathbf{x}$, involving three aspects, namely, language fluency $f_{\overleftrightarrow{\text{LM}}}(\mathbf{y})$, semantic similarity $f_{\text{SIM}}(\mathbf{y}; \mathbf{x})$, and a length constraint $f_{\text{LEN}}(\mathbf{y}, s)$. This is given by

$$f(\mathbf{y}; \mathbf{x}, s) = f_{\overleftrightarrow{\text{LM}}}(\mathbf{y}) \cdot f_{\text{SIM}}(\mathbf{y}; \mathbf{x})^{\gamma} \cdot f_{\text{LEN}}(\mathbf{y}; s), \quad (1)$$

where the relative weight γ balances $f_{\overleftrightarrow{\text{LM}}}(\mathbf{y})$ and $f_{\text{SIM}}(\mathbf{y}; \mathbf{x})$. We treat the summary length as a hard constraint, and therefore we do not need a weighting hyperparameter for f_{LEN}.

Language Fluency. The language fluency scorer quantifies how grammatical and idiomatic a candidate summary $\mathbf{y}$ is. Our model generates a candidate summary in a non-autoregressive fashion, in contrast to the beam search in Zhou and Rush (2019). Thus, we are able to simultaneously consider forward and backward language models, using the geometric average of their perplexities. Using both forward and backward language models is less biased towards sentence beginnings or endings.

$$\overleftrightarrow{\text{PPL}}(\mathbf{y}) = \sqrt[2|\mathbf{y}|]{\prod_{i}^{|\mathbf{y}|} \frac{1}{p_{\overrightarrow{\text{LM}}}(y_i|\mathbf{y}_{<i})} \prod_{i}^{|\mathbf{y}|} \frac{1}{p_{\overleftarrow{\text{LM}}}(y_i|\mathbf{y}_{>i})}}.$$

Our fluency scorer is the inverse perplexity.

$$f_{\overleftrightarrow{\text{LM}}}(\mathbf{y}) = \overleftrightarrow{\text{PPL}}(\mathbf{y})^{-1}. \quad (2)$$

Depending on applications, the language models could be pretrained on a target corpus.[3] In this case, the fluency scorer also measures whether the summary style is consistent with the target language. This could be important in certain applications, e.g., headline generation, where the summary language differs from the input in style.

Semantic Similarity. A semantic similarity scorer ensures that the summary keeps the key information of the input sentence. We adopt the cosine similarity between sentence embeddings as

$$f_{\text{SIM}}(\mathbf{y}; \mathbf{x}) = \cos(\boldsymbol{e}(\mathbf{x}), \boldsymbol{e}(\mathbf{y})), \quad (3)$$

where $\boldsymbol{e}$ is a sentence embedding method. In our work, we use unigram word embeddings learned by the sent2vec model (Pagliardini et al., 2018). Then, $\boldsymbol{e}(\mathbf{x})$ is computed as the average of these unigram embeddings, weighted by the inverse-document frequency (*idf*) of the words.

We use sent2vec because it is trained in an unsupervised way on individual sentences. By contrast, other unsupervised methods like SiameseCBOW (Kenter et al., 2016) or BERT (Devlin et al., 2019) use adjacent sentences as part of the training signal.

Length Constraint. Our discrete searching approach is able to impose the output length as a hard constraint, allowing the model to generate summaries of any given length. Suppose the desired output length is s, then our length scorer is

[3] We use the terminology *unsupervised summarization*, following Zhou and Rush (2019). While we train the language models on the desired target language, we do not need parallel source-target pairs, i.e., sentences together with their groundtruth summaries.

$$f_{\text{LEN}}(\mathbf{y}; s) = \begin{cases} 1, & \text{if } |\mathbf{y}| = s, \\ -\infty, & \text{otherwise.} \end{cases} \quad (4)$$

In other words, a candidate summary $\mathbf{y}$ is *infeasible* if it does not satisfy the length constraint. In practice, we implement this hard constraint by searching among *feasible solutions* only.

3.2 Search Space

Most sentence generation models choose a word from the vocabulary at each time step, such as autoregressive generation that predicts the next word (Sutskever et al., 2014; Rush et al., 2015), and edit-based generation with deletion or insertion operations (Miao et al., 2019; Dong et al., 2019). In these cases, the search space is $|\mathcal{V}|^s$, given a vocabulary $\mathcal{V}$ and a summary length s.

However, reference summaries are highly extractive. In the headline generation dataset (Rush et al., 2015), for example, 45% of the words in the reference summary also appear in the source sentence. This yields a ceiling of 45 ROUGE-1 F1 points[4] for a purely extractive method, which is higher than the current state-of-the-art supervised abstractive result of 39 points (Wang et al., 2019). We are thus motivated to propose our word-extraction approach that extracts a subsequence of the input as the summary. Additionally, we arrange the words in the same order as the input, motivated by the *monotonicity assumption* in summarization (Yu et al., 2016; Raffel et al., 2017).

Formally, we define the search space as $\boldsymbol{a} = (a_1, \ldots, a_n) \in \{0, 1\}^n$, where n is the length of the input sentence $\mathbf{x}$. The vector $\boldsymbol{a}$ is a Boolean filter over the source words $\mathbf{x}$. The summary sequence can then be represented by $\mathbf{y} = \mathbf{x}_{\boldsymbol{a}}$, i.e., we sequentially extract words from the source sequence $\mathbf{x}$ by the Boolean vector $\boldsymbol{a}$. If $a_i = 1$, then x_i is extracted for the summary, and vice versa.

Further, we only consider the search space of all feasible solutions $\{\boldsymbol{a} : f(\mathbf{x}_{\boldsymbol{a}}; \mathbf{x}, s) > -\infty\}$. That is to say, the candidate summary has to satisfy the length constraint in Section 3.1. Equivalently, the output length can be expressed by a constraint on the search space such that $\sum_i a_i = s$.

The above restrictions reduce the search space to $\binom{n}{s}$ solutions. In a realistic setting, our search space is much smaller than that of generating words from the entire vocabulary.

[4]We assume an extracted summary has the same length as the reference, and 45% words of the reference are in the original sentence. This gives us a ceiling of 45% precision and recall.

Algorithm 1 First-Choice Hill Climbing

input objective function $f(\mathbf{y}; \mathbf{x}, s)$, source sentence $\mathbf{x}$, summary length s, number of steps T, initial random solution $\boldsymbol{a}_0$, neighbor function $q(\boldsymbol{a}'|\boldsymbol{a})$
  for $t = 1$ to T **do**
    $\mathbf{y}_{t-1} = \mathbf{x}_{\boldsymbol{a}_{t-1}}$
    $\boldsymbol{a}' \sim q(\cdot|\boldsymbol{a}_{t-1})$
    $\mathbf{y}' = \mathbf{x}_{\boldsymbol{a}'}$
    if $f(\mathbf{y}'; \mathbf{x}, s) \geq f(\mathbf{y}_{t-1}; \mathbf{x}, s)$ **then**
      $\boldsymbol{a}_t = \boldsymbol{a}'$
    else
      $\boldsymbol{a}_t = \boldsymbol{a}_{t-1}$
  return $\mathbf{y}^* \leftarrow \mathbf{x}_{\boldsymbol{a}_T}$

3.3 Search Algorithm

We optimize our objective function $f(\mathbf{y}; \mathbf{x}, s)$ by first-choice hill climbing (FCHC, Russell and Norvig, 2016). This is a stochastic optimization algorithm that proposes a candidate solution by local change at every search step. The candidate is accepted if it is better than the current solution. Otherwise, the algorithm keeps the current solution. FCHC maximizes the objective function in a greedy fashion and yields a (possibly local) optimum.

Algorithm 1 shows the optimization procedure of our FCHC. For each search step, a new candidate is sampled from the neighbor function $q(\boldsymbol{a}'|\boldsymbol{a})$. This is accomplished by randomly swapping two actions a_i and a_j for $a_i \neq a_j$, i.e., replacing a word in the summary with a word from the source sentence that is not in the current summary. The order of selected words is kept as in the source sentence. If the candidate solution achieves a higher score, then it is accepted. Otherwise, the candidate is rejected and the algorithm proceeds with the current solution. Our search terminates if it exceeds a predefined budget. The last solution is returned as the summary, as it is also the best-scored candidate due to our greedy algorithm.

One main potential drawback of hill climbing algorithms is that they may get stuck in a local optimum. To alleviate this problem, we restart the algorithm with multiple random initial word selections $\boldsymbol{a}_0$ and return the overall best solution. We set the number of restarts as $\beta_R \cdot ns^2$ and number of search steps as $\beta_T \cdot ns^2$, where β_R and β_T are controlling hyperparameters. We design the formula to encourage more search for longer input sentences, but only with a tractable growth: linear for input length and quadratic for summary length. As the summary length is usually much smaller

than the input length, quadratic search is possible. Increasing the number of restarts (and search steps) monotonically improves the scoring function, and thus in practice can be set according to the available search budget.

Other discrete optimization algorithms can be explored for sentence generation, such as simulated annealing (Liu et al., 2020) and genetic algorithms. Our analysis on short sentences (where exhaustive search is tractable) showed that hill climbing with restarts achieves ROUGE scores similar to exhaustive search (Section 5.4).

4 Evaluation Framework

In this section, we will describe the datasets, evaluation metrics, and a widely used baseline (called Lead). Additionally, we report the observation that the commonly used evaluation metric, ROUGE F1, is sensitive to summary length, preferring longer summaries. Thus, we propose to group models with similar output length during evaluation for fair comparison.

4.1 Datasets

We evaluate our models on the dataset provided for DUC2004 Task 1 (Over and Yen, 2004) and a headline generation corpus[5] (Rush et al., 2015), both widely adopted in the summarization literature.

The DUC2004 dataset is designed and used for testing only. It consists of 500 news articles, each paired with four human written summaries. We follow Rush et al. (2015) and adopt DUC2004 for sentence summarization by using only the first sentence of an article as input. The reference summaries are around 10 words long on average.

The headline generation dataset (Rush et al., 2015) is derived from the Gigaword news corpus. Each headline/title is viewed as the reference summary of the first sentence of an article. The dataset contains 3.8M training instances and 1951 test instances. The average headline contains ∼8 words; the average source sentence contains ∼30 words. We use 500 held-out validation instances for hyperparameter tuning. Note that the training set is only used to train a language model and sent2vec embeddings. The summarization process itself is not trained in our approach.

[5] https://github.com/harvardnlp/NAMAS

4.2 Lead Baselines

Lead baselines are a strong competitor that extracts the first few characters or words of the input sentence. The DUC2004 shared task includes a Lead baseline, which extracts the first 75 characters as the summary. We call it Lead-C-75. For the Gigaword dataset, the reference has 8 words on average, and it is common to compare with a Lead variant that chooses the first 8 words. We call this baseline Lead-N-n when we choose n words. For fair comparison with previous work (Baziotis et al., 2019; Fevry and Phang, 2018) in Section 5.2, we further introduce a new variant that returns the first p percent of source words as the summary. We denote this baseline by Lead-P-p.

4.3 ROUGE Scores

Summarization systems are commonly evaluated by ROUGE scores (Lin, 2004). The ROUGE-1 (or ROUGE-2) score computes the unigram (or bigram) overlap of a generated summary and the reference. ROUGE-L calculates the longest common subsequence. Depending on the dataset, either ROUGE Recall or ROUGE F1 variant is adopted. Since the ROUGE Recall metric is not normalized with regard to length, DUC2004 standard evaluation truncates the summary at 75 characters. This procedure was also adopted by Rush et al. (2015) for the headline generation task, but later Chopra et al. (2016) proposed to report the "more balanced" ROUGE F1 metric for the Gigaword headline generation dataset and abandoned truncation. We follow previous work and use ROUGE F1 for headline generation and truncated ROUGE Recall for DUC2004.

4.4 Summary Length

As mentioned, ROUGE F1 was introduced to the evaluation of sentence summarization to better compare models with different output lengths (Chopra et al., 2016; Nallapati et al., 2016). To investigate the effect of summary length on ROUGE F1, we calculate ROUGE F1 scores for the Lead-N-n and Lead-P-p baselines with different length parameters. Figure 2 shows that ROUGE F1 peaks at $n \approx 18$ or $p \approx 50$. The difference between the maximum performance at $n \approx 18$ and the widely adopted baseline (Lead-N-8) is large: 4.2 ROUGE-1 F1 points. A similar effect is observed by Sun et al. (2019) for document summarization. This shows that ROUGE F1 is still sensitive to summary length, and this effect should be

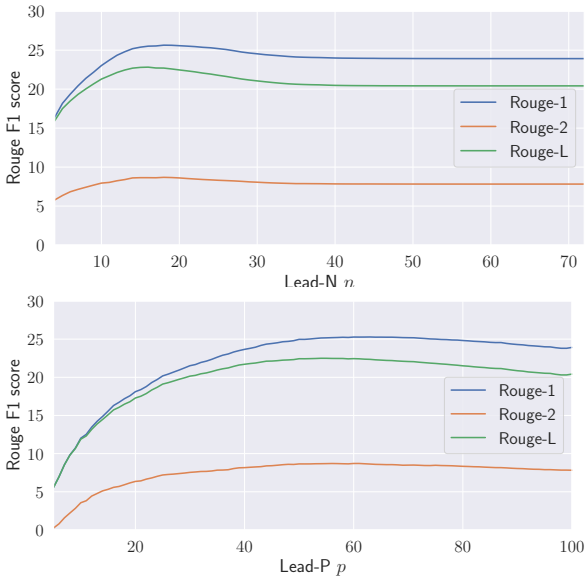

Figure 2: ROUGE F1 scores on the test set of headline generation for Lead-N and Lead-P baselines with different number n and percentage p of leading words.

considered during evaluation. We propose to report the average output length of a model and only compare models in the same length bracket.

5 Experiments

5.1 Setup

We conduct experiments with two settings, dependent on how the scorers $f_{\overleftrightarrow{\text{LM}}}$ and f_{SIM} are trained. In the first setting, we train the language model and sent2vec embeddings on the **source** (article) side of the Gigaword headline generation dataset. This complies with Fevry and Phang (2018) and Baziotis et al. (2019). In the second setting, we train the language model and sent2vec embeddings on the **target** (title) side like Zhou and Rush (2019). In both settings, we do not need parallel source-target pairs.

For output length, our headline generation experiment sets the desired target length as 8 words, 10 words, and 50% of the input, as these mirror either the average reference summary length or the average output lengths of our competitors (Wang and Lee, 2018; Zhou and Rush, 2019; Fevry and Phang, 2018; Baziotis et al., 2019). For DUC2004, the desired summary length is set to 13 words, because the standard evaluation script truncates after the first 75 characters (roughly 13 words) in the summary.

Our forward and backward language models use long short term memory units (Hochreiter and Schmidhuber, 1997) and are optimized for 50 epochs by stochastic gradient descent. Embeddings and hidden sizes are set to 1024 dimensions.

We tune hyperparameters on the development data of the headline corpus, and set the weighting parameter γ to 12 for all models. The search steps and restarts are set to $\beta_T = 0.1$ and $\beta_R = 0.035$, respectively. We see a sharp performance improvement when we do more searching. Thus, we choose β_T and β_R at the critical values due to efficiency concerns.

5.2 Competing Models

Besides the Lead baselines discussed in Section 4.2, we compare our models with state-of-the-art unsupervised sentence summarization systems.

Wang and Lee (2018)[6] use cycle-consistency to reconstruct source sentences from the headline generation corpus (Rush et al., 2015). The latent discrete representation, learned to be similar to (non-parallel) headlines, is used as the summary.

Zhou and Rush (2019) optimize an objective function involving language fluency and contextual matching. Their language modeling scorer is trained on headlines of the Gigaword training set; their contextual matching scorer is based on ELMo embeddings (Peters et al., 2018) trained with the Billion Word corpus (Chelba et al., 2013). Their summary length is controlled by a soft length penalty during beam search.

Fevry and Phang (2018)[7] learn a denoising autoencoder (Vincent et al., 2008) to reconstruct source sentences of the Gigaword training set. Summary length is set to 50% of the input length and is controlled by length embeddings in the decoder.

Baziotis et al. (2019)[8] propose SEQ^3 that uses cycle-consistency to reconstruct source sentences from the Gigaword training set. The length is also set to 50% of the input length, controlled by length embeddings in the intermediate decoder.

For the DUC2004 dataset, TOPIARY (Zajic et al., 2004) is the winning system in the competition. They shorten the sentence by rule-based syntax-tree trimming (Dorr et al., 2003), but enhance the resulting summary with topics that are learned on

[6]Generated summaries are obtained via E-Mail correspondence. Scores differ because of evaluation setup.

[7]Retrained with official code (https://github.com/zphang/usc_dae) because the authors use a private test set.

[8]Retrained with official code (https://github.com/cbaziotis/seq3), because of different test data. The authors remove 54 noisy instances. Our replication thus achieves slightly lower scores than theirs.

	Model	Data			Len D	ROUGE F1			Len O
		article	title	external		R-1	R-2	R-L	
	Lead-N-8	✓			8	21.39	7.42	20.03	7.9
A	*HC_article_8*	✓			8	23.09	7.50	21.29	7.9
	HC_title_8		✓		8	**26.32**	**9.63**	**24.19**	7.9
	Lead-N-10	✓			10	23.03	7.95	21.29	9.8
	Wang and Lee (2018)	✓	✓		-	27.29	10.01	24.59	10.8
	Zhou and Rush (2019)		✓	billion	-	26.48	10.05	24.41	9.3
B	*HC_article_10*	✓			10	24.44	8.01	22.21	9.8
	HC_title_10		✓		10	27.52	10.27	24.91	9.8
	HC_title+twitter_10		✓	twitter	10	28.26	10.42	25.43	9.8
	HC_title+billion_10		✓	billion	10	**28.80**	**10.66**	**25.82**	9.8
	Lead-P-50	✓			50%	24.97	8.65	22.43	14.6
	Fevry and Phang (2018)	✓		SNLI	50%	23.16	5.93	20.11	14.8
C	Baziotis et al. (2019)	✓			50%	24.70	7.97	22.14	15.1
	HC_article_50p	✓			50%	25.58	8.44	22.66	14.9
	HC_title_50p		✓		50%	**27.05**	**9.75**	**23.89**	14.9

Table 1: Results for headline generation on the Gigaword test set. **Data**: data used during training (source article, target titles, external corpus). *billion*: the Billion Word Corpus (Chelba et al., 2013); *twitter*: the Twitter corpus (Pagliardini et al., 2018); *SNLI*: the Stanford Natural Language Inference dataset (Bowman et al., 2015). **Len D**: desired summary length. **ROUGE F1 (R-1, R-2, R-L)**: ROUGE-1, ROUGE-2, ROUGE-L F1 scores. **Len O**: averaged output length. **Best** results in bold. Second best results underlined. **A**: Models with output length around 8 words. **B**: Models with output length around 10 words. **C**: Models with output length around 50% of the input. Our hill-climbing (HC) approaches are named in the format of *HC_data_outputLength*.

Model	ROUGE Recall		
	R-1	R-2	R-L
Lead-C-75	22.50	6.49	19.72
SEQ3 (Baziotis et al., 2019)	22.13	6.18	19.3
TOPIARY (Zajic et al., 2004)	25.12	6.46	20.12
BOTTLESUM EX (West et al., 2019)	22.85	5.71	19.87
HC_article_13	24.21	6.63	21.24
HC_title_13	26.04	8.06	22.90
HC_title+twitter_13	**27.41**	**8.76**	**23.89**

Table 2: Results on the DUC2004 dataset.

full articles.

BOTTLESUM EX (West et al., 2019) uses the information bottleneck principle to predict the next sentence in an article. Their method employs a pretrained small GPT-2 model (Radford et al., 2019).

5.3 Results

Results for Headline Generation. We first compare with Lead-N-8 (Group A, Table 1). This is a standard baseline in previous work, because the average reference summary contains eight words. Unfortunately, none of the previous papers consider output length during evaluation, making comparisons between their (longer) output summaries and the Lead-N-8 baseline unfair, as discussed in Section 4.4. Our approach, which explicitly controls summary length, considerably outperforms the Lead-N-8 baseline in a fair setting.

Next, we compare with state-of-the-art unsupervised methods, whose output summary has roughly 10 words on average (Group B). In this case, we set our hard length constraint as 10 and include the Lead-N-10 baseline for comparison. Trained on the title side only, our *HC_title_10* model outperforms these competing methods in all ROUGE F1 scores. In particular, Zhou and Rush (2019) use the target side to train the language model, plus the Billion Word Corpus to pretrain embeddings used in the contextual matching scorer. With the same extra corpus to pretrain our sent2vec embeddings, our *HC_title+billion_10* variant achieves even better performance, outperforming Zhou and Rush (2019) by 2.32 ROUGE-1 and 1.41 ROUGE-L points.

The Billion Word Corpus, however, includes complete articles, which implicitly yields unaligned parallel data. This could be inappropriate for an unsupervised method. Thus, we further train sent2vec embeddings on the Twitter corpus by Pagliardini et al. (2018). The *HC_title+twitter_10* also performs better than *HC_title_10* and other competitors.

In Group C, we compare with the models whose summaries have an average length of 50% of the input sentence. We set our desired target length to 50% as well, and include the Lead-P-50 baseline. Previous studies report a performance improvement over the Lead-N-8 baseline, but in fact, Table 1 shows that they do not outperform the appropriate Lead baseline Lead-P-50. Our model is the only unsupervised summarization system that outperforms the Lead-P-50 baseline on this dataset,

even though it is trained solely on the article side.

It is noted that our models trained on the title side (*HC_title*) consistently outperform those trained on the article side (*HC_article*). This is not surprising because the former can generate headlines from the learned target distribution. This shows the importance of learning a summary language model even if we do not have supervision of parallel source-target data.

Results for DUC2004. Table 2 shows the results on the DUC2004 data. As this dataset is for test only, we directly transfer the models *HC_article* and *HC_title* from the headline generation corpus with the same hyperparameters (except for length). As shown in the table, we outperform all previous methods and the Lead-C-75 baseline. The results are consistent with Table 1, showing the generalizability of our approach.

Human Evaluation. We conduct human evaluation via pairwise comparison of system outputs, in the same vein as (West et al., 2019). The annotator sees the source sentence along with the headline generated by our system and a competing method, presented in random order. The annotator is asked to compare the fidelity and fluency of the two systems, choosing among the three options (i) the first headline is better (ii) the second headline is better, and (iii) both headlines are equally good/bad. This task is repeated for 100 instances with 5 annotators each. The final label is selected by majority voting. The inter-annotator agreement (Krippendorff's alpha) is 0.25 when our model is compared with Wang and Lee (2018) and 0.17 with Zhou and Rush (2019).

We report the aggregated score of our system in Table 3. For each sample, we count 1 point if our model wins, 0 points if it ties, -1 point if it loses. The points are normalized by the number of samples. The results show an advantage of our model over Wang and Lee (2018), especially in fluency. Our model is also on par with Zhou and Rush (2019). Note again that we achieve this with fewer data.

5.4 Analysis

In this section, we conduct an in-depth analysis of our model, based on *HC_title_10* for headline generation.

Search Objective. Table 4 provides an ablation study on our objective function. It shows that both language fluency and semantic similarity play a role in measuring the quality of a summary. The bi-directional language model is also slightly better than a uni-directional language model.

Models	Score (#wins/#ties/#loses)	
	Fidelity	Fluency
HC vs. WL	+0.18 (44/30/26)	+0.30 (45/40/15)
HC vs. ZR	+0.05 (35/35/30)	-0.03 (24/49/27)

Table 3: Human evaluation in a pairwise comparison setting on 100 headline generation instances. We show the scores of our model (*HC_title_10*) when it is compared with WL (Wang and Lee, 2018) and ZR (Zhou and Rush, 2019), in terms of average score of fidelity and fluency: 1 (wins), 0 (ties), and -1 (loses).

Objective	ROUGE F1 scores		
$f =$	R-1	R-2	R-L
$f_{\overleftrightarrow{\text{LM}}} \cdot f_{\text{SIM}}$ (full model)	**27.52**	**10.27**	**24.91**
$f_{\overrightarrow{\text{LM}}} \cdot f_{\text{SIM}}$	27.50	10.15	24.79
$f_{\overleftrightarrow{\text{LM}}}$	25.24	8.87	23.09
$f_{\overrightarrow{\text{LM}}}$	25.18	8.72	22.93
f_{SIM}	20.31	4.08	18.19

Table 4: Ablation study of the search objective. Model *HC_title_10* on the headline generation test set. Length constraint term omitted from notation.

Search Algorithm. In Figure 3, we compare our FCHC with the theoretical optimum on short sentences where exhaustive search is tractable. For only 3% of the instances with source sentence length between 25 and 30 words, our FCHC algorithm does not find the global optimum. In 21% of those cases, the better objective score leads to a higher ROUGE-L score. This shows that FCHC with restarts is a powerful enough search algorithm for word extraction-based sentence summarization.

Positional Bias. We analyze the positional bias of each algorithm by plotting the normalized frequency of extracted words within four different areas of the source sentence. As shown in Figure 4, the extraction positions of words in the reference headlines are slightly skewed towards the beginning of the source sentence. Our hill-climbing algorithm performs distributed edits over the sentence, which is reflected in the flatter graph across the source sentence areas. By contrast, beam search (Zhou and Rush, 2019) is more biased towards the first quarter of the source sentence. Cycle consistency models (Wang and Lee, 2018; Baziotis et al., 2019) show a strong bias towards the first half of the source sentence. We suspect that the reconstruction decoder is easily satisfied with the beginning of the source sentence as the discrete latent variable,

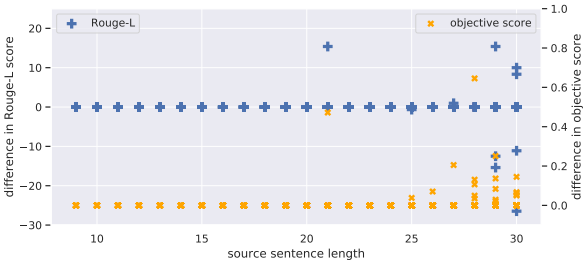

Figure 3: Orange crosses show the objective score optimized by exhaustive search minus the objective score optimized by FCHC. Blue pluses show the ROUGE-L difference between exhaustive search and FCHC. Plotted for the 1135 instances in the headline generation test set, where the source sentence has 30 words or fewer.

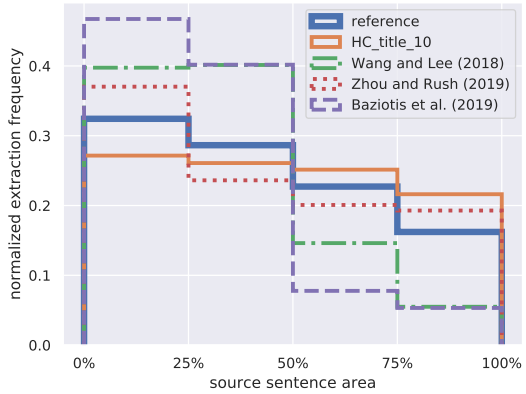

Figure 4: Positional bias for different systems, calculated for the headline generation test set. The source sentence is divided into four areas: 0–25%, 25–50%, 50–75%, and 75-100% of the sentence. The y-axis shows the normalized frequency of how often a word in the summary is extracted from one of the four source sentence areas.

because of its autoregressive decoding.

Case Study. We show example summaries generated by our system in Figure 5. We see that the *HC_title* models indeed learn the style of headlines, known as *headlinese*. As shown, *HC_title* often uses simple tense and drops articles (e.g., "a" and "the"). The summaries generated by *HC_article* tend to waste word slots by including an uninformative determiner.

It is also seen that we can control the length in an explicit way. Comparing *HC_title* with desired lengths of 8 and 10, we see that the additional two words are used to include more information, such as the day of the meeting in Example 2 or the gender of the injured person in Example 3.

1. Input: a german registered container ship ran aground at the entrance to the french port of le havre early tuesday , but authorities said there were no casualties . **Reference**: container ship runs aground in french port ***HC_article_10***: a container ship ran aground but there were no casualties ***HC_title_10***: container ship ran aground at french port but no casualties ***HC_title_8***: ship ran aground at french port no casualties
2. Input: fidel castro , cuba's president of the council of state , met with a chinese delegation here tuesday . **Reference:** castro meets chinese official ***HC_article_10***: fidel castro cuba 's president met with a chinese delegation ***HC_title_10***: fidel castro cuba 's president met with chinese delegation tuesday ***HC_title_8***: fidel castro 's president met with chinese delegation
3. Input: two grenades exploded near a national police station monday , slightly injuring one woman , news reports said . **Reference:** two grenades explode near spanish police station ***HC_article_10***: two grenades exploded near a police station injuring one woman ***HC_title_10***: two grenades exploded near a police station injuring one woman ***HC_title_8***: two grenades exploded near police station injuring one

Table 5: Example summaries for headline generation test set.

6 Conclusion

We proposed a novel word-extraction model for sentence summarization that generates summaries by optimizing an objective function of language fluency and semantic similarity. A hard length constraint is also imposed in our objective function. In a controlled experiment, our model achieves better performance than strong baselines on headline generation and DUC2004 datasets.

Acknowledgments

We acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC), under grant Nos. RGPIN-2019-04897, and RGPIN-2020-04465. Lili Mou is also supported by AltaML, the Amii Fellow Program, and the Canadian CIFAR AI Chair Program. This research was enabled in part by the support of Compute Canada (www.computecanada.ca).

References

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2017. Guided open vocabulary image captioning with constrained beam search. In *EMNLP*, pages 936–945.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Ben-

gio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

Christos Baziotis, Ion Androutsopoulos, Ioannis Konstas, and Alexandros Potamianos. 2019. SEQ^3: Differentiable sequence-to-sequence-to-sequence autoencoder for unsupervised abstractive sentence compression. In *NAACL-HLT*, pages 673–681.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*, pages 632–642.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2013. One billion word benchmark for measuring progress in statistical language modeling. *arXiv preprint arXiv:1312.3005*.

Sumit Chopra, Michael Auli, and Alexander M. Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *NAACL-HLT*, pages 93–98.

James Clarke and Mirella Lapata. 2006. Constraint-based sentence compression an integer programming approach. In *COLING-ACL*, volume 2, pages 144–151.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186.

Yue Dong, Zichao Li, Mehdi Rezagholizadeh, and Jackie Chi Kit Cheung. 2019. EditNTS: An neural programmer-interpreter model for sentence simplification through explicit editing. In *ACL*, pages 3393–3402.

Bonnie Dorr, David Zajic, and Richard Schwartz. 2003. Hedge trimmer: A parse-and-trim approach to headline generation. In *HLT-NAACL Workshop on Text Summarization*, pages 1–8.

Günes Erkan and Dragomir R. Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *JAIR*, 22(1):457–479.

Alexander R Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir R Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In *ACL*, pages 1074–1084.

Angela Fan, David Grangier, and Michael Auli. 2018. Controllable abstractive summarization. In *Proc. 2nd Workshop on Neural Machine Translation and Generation*, pages 45–54.

Thibault Fevry and Jason Phang. 2018. Unsupervised sentence compression using denoising auto-encoders. In *CoNLL*, pages 413–422.

Sebastian Gehrmann, Yuntian Deng, and Alexander Rush. 2018. Bottom-up abstractive summarization. In *EMNLP*, pages 4098–4109.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *ACL*, pages 1631–1640.

Aria Haghighi and Lucy Vanderwende. 2009. Exploring content models for multi-document summarization. In *HLT-NAACL*, pages 362–370.

Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tieyan Liu, and Wei-Ying Ma. 2016. Dual learning for machine translation. In *NIPS*, pages 820–828.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *ACL*, pages 1535–1546.

Chiori Hori and Sadaoki Furui. 2004. Speech summarization: An approach through word extraction and a method for evaluation. *IEICE Trans. Inf. & Syst.*, 87(1):15–25.

Tom Kenter, Alexey Borisov, and Maarten de Rijke. 2016. Siamese CBOW: Optimizing word embeddings for sentence representations. In *ACL*, pages 941–951.

Yuta Kikuchi, Graham Neubig, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. 2016. Controlling output length in neural encoder-decoders. In *EMNLP*, pages 1328–1338.

Dhruv Kumar, Lili Mou, Lukasz Golab, and Olga Vechtomova. 2020. Iterative edit-based unsupervised sentence simplification. In *ACL*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *ACL Workshop: Text Summarization Branches Out*, pages 74–81.

Xianggen Liu, Lili Mou, Fandong Meng, Hao Zhou, Jie Zhou, and Sen Song. 2020. Unsupervised paraphrasing by simulated annealing. In *ACL*.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2019. CGMH: Constrained sentence generation by Metropolis-Hastings sampling. In *AAAI*, pages 6834–6842.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *EMNLP*, pages 404–411.

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

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *CoNLL*, pages 280–290.

P Over and J Yen. 2004. An introduction to DUC-2004: Intrinsic evaluation of generic news text summarization systems. In *Proc. Document Understanding Conference*.

Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. 2018. Unsupervised learning of sentence embeddings using compositional n-gram features. In *NAACL-HLT*, pages 528–540.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *ICLR*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL-HLT*, pages 2227–2237.

Dragomir R. Radev, Hongyan Jing, and Malgorzata Budzikowska. 2000. Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies. In *NAACL-ANLP 2000 Workshop: Automatic Summarization*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*.

Colin Raffel, Minh-Thang Luong, Peter J Liu, Ron J Weiss, and Douglas Eck. 2017. Online and linear-time attention by enforcing monotonic alignments. In *ICML*, pages 2837–2846.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *EMNLP*, pages 379–389.

Stuart J Russell and Peter Norvig. 2016. *Artificial Intelligence: A Modern Approach*. Pearson Education Limited.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *ACL*, pages 1073–1083.

Simeng Sun, Ori Shapira, Ido Dagan, and Ani Nenkova. 2019. How to compare summarizers without target length? Pitfalls, solutions and re-examination of the neural summarization literature. In *Proc. Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 21–29.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*, pages 3104–3112.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *ICML*, pages 1096–1103.

Kai Wang, Xiaojun Quan, and Rui Wang. 2019. BiSET: Bi-directional selective encoding with template for abstractive summarization. In *ACL*, pages 2153–2162.

Yaushian Wang and Hung-yi Lee. 2018. Learning to encode text as human-readable summaries using generative adversarial networks. In *EMNLP*, pages 4187–4195.

Peter West, Ari Holtzman, Jan Buys, and Yejin Choi. 2019. BottleSum: Unsupervised and self-supervised sentence summarization using the information bottleneck principle. In *EMNLP-IJCNLP*, pages 3750–3759.

Lei Yu, Jan Buys, and Phil Blunsom. 2016. Online segment to segment neural transduction. In *EMNLP*, pages 1307–1316.

David Zajic, Bonnie Dorr, and Richard Schwartz. 2004. BBN/UMD at DUC-2004: Topiary. In *HLT-NAACL Document Understanding Workshop*, pages 112–119.

Hao Zheng and Mirella Lapata. 2019. Sentence centrality revisited for unsupervised summarization. In *ACL*, pages 6236–6247.

Jiawei Zhou and Alexander M Rush. 2019. Simple unsupervised summarization by contextual matching. In *ACL*, pages 5101–5106.

Liang Zhou and Eduard Hovy. 2004. Template-filtered headline summarization. In *ACL Workshop: Text Summarization Branches Out*, pages 56–60.