BottleSum: Unsupervised and Self-supervised Sentence Summarization using the Information Bottleneck Principle

Peter West[1] **Ari Holtzman**[1,2] **Jan Buys**[1] **Yejin Choi**[1,2]
[1]Paul G. Allen School of Computer Science & Engineering, University of Washington
[2]Allen Institute for Artificial Intelligence
{pawest, ahai, jbuys, yejin}@cs.washington.edu

Abstract

The principle of the Information Bottleneck (Tishby et al., 1999) is to produce a summary of information X optimized to predict some other relevant information Y. In this paper, we propose a novel approach to unsupervised sentence summarization by mapping the Information Bottleneck principle to a conditional language modelling objective: given a sentence, our approach seeks a compressed sentence that can best predict the *next* sentence. Our iterative algorithm under the Information Bottleneck objective searches gradually shorter subsequences of the given sentence while maximizing the probability of the next sentence conditioned on the summary. Using only pretrained language models with no direct supervision, our approach can efficiently perform extractive sentence summarization over a large corpus.

Building on our *unsupervised extractive* summarization (*BottleSum*^{Ex}), we then present a new approach to *self-supervised abstractive* summarization (*BottleSum*^{Self}), where a transformer-based language model is trained on the output summaries of our unsupervised method. Empirical results demonstrate that our extractive method outperforms other unsupervised models on multiple automatic metrics. In addition, we find that our self-supervised abstractive model outperforms unsupervised baselines (including our own) by human evaluation along multiple attributes.

1 Introduction

Recent approaches based on neural networks have brought significant advancements for both extractive and abstractive summarization (Rush et al., 2015; Nallapati et al., 2016). However, their success relies on large-scale parallel corpora of input text and output summaries for direct supervision. For example, there are ~280,000 training instances in the CNN/Daily Mail dataset (Nallapati et al., 2016; Hermann et al., 2015), and ~4,000,000 instances in the sentence summarization dataset of Rush et al. (2015). Because it is too costly to have humans write gold summaries at this scale, existing large-scale datasets are based on naturally occurring pairs of summary-like text paired with source text, for instance using news titles or highlights as summaries for news-text. A major drawback to this approach is that these pairs must already exist in-domain, which is often not true.

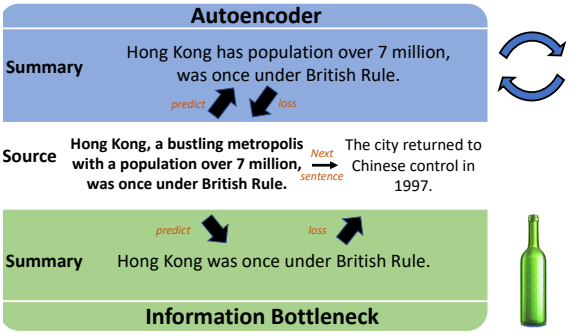

Figure 1: Example contrasting the Autoencoder (AE) and Information Bottleneck (IB) approaches to summarization. While AE (top) preserves any detail that helps to reconstruct the original, such as population size in this example, IB (bottom) uses context to determine which information is relevant, which results in a more appropriate summary.

The sample inefficiency of current neural approaches limits their impact across different tasks and domains, motivating the need for unsupervised or self-supervised alternatives (Artetxe et al., 2017; LeCun, 2018; Schmidhuber, 1990). Further, for summarization in particular, the current paradigm requiring millions of supervision examples is almost counter-intuitive; after all, humans don't need to see a million summaries to know how to summarize, or what information to include.

In this paper, we present *BottleSum*, consisting

of a pair of novel approaches, *BottleSum*^{Ex} and *BottleSum*^{Self} for *unsupervised extractive* and *self-supervised abstractive* summarization, respectively. Core to our approach is the principle of the Information Bottleneck (Tishby et al., 1999), producing a summary for information X optimized to predict some other relevant information Y. In particular, we map (conditional) language modeling objectives to the Information Bottleneck principle to guide the unsupervised model on what to keep and what to discard.

The key intuition of our bottleneck-based summarization is that a good sentence summary contains information related to the broader context while discarding less significant details. Figure 1 demonstrates this intuition. Given input sentence "Hong Kong, a bustling metropolis with a population over 7 million, ...", which is followed by the next sentence "The city returned to Chinese control in 1997", the information bottleneck would suggest that minute details such as the city's population being over 7 million are relatively less important to keep. In contrast, the continued discussion of the city's governance in the next sentence suggests its former British rule is important here.

This intuition contrasts with that of autoencoder-based approaches where the goal is to minimize the reconstruction loss of the input sentence when constructing the summary (Miao and Blunsom, 2016; Wang and Lee, 2018; Fevry and Phang, 2018; Baziotis et al., 2019). Under the reconstruction loss, minute but specific details such as the city's population being over 7 million will be difficult to discard from the summary, because they are useful for reconstruction.

Concretely, *BottleSum*^{Ex} is an extractive and unsupervised sentence summarization method using the next sentence, a sample of nearby context, as guidance to *relevance*, or what information to keep. We capture this with a conditional language modelling objective, allowing us to benefit from powerful deep neural language models that are pre-trained over an extremely large-scale corpus. Under the Information Bottleneck objective, we present an iterative algorithm that searches gradually shorter subsequences of the source sentence while maximizing the probability of the next sentence conditioned on the summary. The benefit of this approach is that it requires no domain-specific supervision or fine-turning.

Building on our unsupervised extractive summarization, we then present *BottleSum*^{Self}, a new approach to self-supervised abstractive summarization. This method also uses a pretrained language model, but turns it into an abstractive summarizer by fine-tuning on the output summaries generated by *BottleSum*^{Ex} paired with their original input sentences. The goal is to generalize the summaries generated by an extractive method by training a language model on them, which can then produce abstractive summaries as its generation is not constrained to be extractive.

Together, *BottleSum*^{Ex} and *BottleSum*^{Self} are *BottleSum* methods for unsupervised sentence summarization. Empirical results demonstrate that *BottleSum*^{Ex} outperforms other unsupervised methods on multiple automatic metrics, closely followed by *BottleSum*^{Self}. Furthermore, testing on a large unsupervised corpus, we find *BottleSum*^{Self} outperforms unsupervised baselines (including our own *BottleSum*^{Ex}) on human evaluation along multiple attributes.

2 The Information Bottleneck Principle

Unsupervised summarization requires formulating an appropriate learning objective that can be optimized without supervision (example summaries). Recent work has treated unsupervised summarization as an autoencoding problem with a reconstruction loss (Miao and Blunsom, 2016; Baziotis et al., 2019). The goal is then to produce a compressed summary from which the source sentence can be accurately predicted, i.e. to maximize:

$$\mathbb{E}_{p(\tilde{s}|s)} \ \log p(s|\tilde{s}), \tag{1}$$

where s is the source sentence, $\tilde{s}$ is the generated summary and $p(\tilde{s}|s)$ the learned summarization model. The exact form of this loss may be more elaborate depending on the system, for example including an auxiliary language modeling loss, but the main aim is to produce a summary from which the source can be reconstructed.

The intuitive limitation of this approach is that it will always prefer to retain all informative content from the source. This goes against the fundamental goal of summarization, which crucially needs to forget all but the "relevant" information. It should be detrimental to keep tangential information, as illustrated by the example in Figure 1. As a result, autoencoding systems need to introduce additional loss terms to augment the reconstruction loss (e.g. length penalty, or the topic loss

of Baziotis et al. (2019)).

The premise of our work is that the Information Bottleneck (IB) principle (Tishby et al., 1999) is a more natural fit for summarization. Unlike reconstruction loss, which requires augmentative terms to summarize, IB naturally incorporates a tradeoff between information selection and pruning. These approaches are compared directly in section 5.

At its core, IB is concerned with the problem of maximal compression while defining a formal notion of information relevance. This is introduced with an external variable Y. The key is that $\tilde{S}$, the summary of source S, contains only information useful for predicting Y. This can be posed formally as learning a conditional distribution $p(\tilde{S}|S)$ minimizing:

$$I(\tilde{S};S) - \beta I(\tilde{S};Y), \tag{2}$$

where I denotes mutual information between these variables.

A notion of information relevance comes from the second term, the *relevance term*: with a positive coefficient β, this is encouraging summaries $\tilde{S}$ to contain information shared with Y. The first term, or *pruning term*, ensures that irrelevant information is discarded. By minimizing the mutual information between summary $\tilde{S}$ and source S, any information about the source that is not credited by the *relevance term* is thrown away. The statistical structure of IB makes this compressive by forcing the summary to only contain information shared with the source.[1]

In sum, IB relies on 3 principles:

1. Encouraging relevant information with a *relevance term*.

2. Discouraging extra information with a *pruning term*.

3. Strictly summarizing the source.

To clarify the difference from a reconstructive loss, suppose there is irrelevant information in S (i.e. unrelated to relevance variable Y), call this Z. With the IB objective (eq 3), there is no benefit to keeping any information from Z, which strictly makes the first term worse (more mutual information between source and summary) and does not affect the second (Z is unrelated to Y). In contrast, because Z contains information about S, including it in $\tilde{S}$ could easily benefit the reconstructive loss (eq. 1) despite being irrelevant.

As a relevance variable we will use the sentence following the source in the document in which it occurs. This choice is motivated by linguistic cohesion, in which we expect more broadly relevant information to be common between consecutive sentences, while less relevant information and details are often not carried forward.

We use these principles to derive two methods for sentence summarization. Our first method (§3) enforces strict summarization through being extractive. Additionally, it does not require any training, so can be applied directly without the availability of domain-specific data. The second method (§4) generalizes IB-based summarization to *abstractive* summarization that can be trained on large unsupervised datasets, learning an explicit summarization function $p(\tilde{s}|s)$ over a distribution of inputs.

3 Unsupervised Extractive Summarization

We now use the Information Bottleneck principle to propose *BottleSum*^{Ex}, an unsupervised extractive approach to sentence summarization. Our approach does not require any training; only a pretrained language model is required to satisfy the IB principles of (2), and the stronger the language model, the stronger our approach will be. In section 5 we demonstrate the effectiveness of this method using GPT-2, the pretrained language model of Radford et al. (2019). [2]

3.1 IB for Extractive Summarization

Here, we take advantage of the natural parallel between the Information Bottleneck and summarization developed in section 2. Working from the 3 IB principles stated there, we derive a set of actionable principles for a concrete sentence summarization method.

We approach the task of summarizing a single sentence s using the following sentence s_{next} as the relevance variable. The method will be a deterministic function mapping s to the summary $\tilde{s}$, so instead of learning a distribution over summaries, we take $p(\tilde{s}|s) = 1$ for the summary we arrive at. Our goal is then to optimize the IB equation (Eq 2)

[1] In IB, this is a strict statistical relationship.

[2] We use the originally released "small" 117M parameter version.

for a single example rather a distribution of inputs (as in the original IB method).

In this setting, to minimize equation 2 we can equivalently minimize:

$$-\log p(\tilde{s}) - \beta_1 p(s_{next}|\tilde{s}) p(\tilde{s}) \log p(s_{next}|\tilde{s}), \quad (3)$$

where coefficient $\beta_1 > 0$ controls the trade-off between keeping relevant information and pruning. See appendix A for the derivation of this equation. Similar to eq 2, the first term encourages pruning, while the second encourages information about the relevance variable, s_{next}. Both unique values in eq 3 ($p(\tilde{s})$ and $p(s_{next}|\tilde{s})$) can be estimated directly by a pretrained language model, a result of the summary being natural language as well as our choice of relevance variable. This will give us a direct path to enforcing IB principles 1 and 2 from section 2.

To interpret principle 3 for text, we consider what attributes are important to strict textual summarization. Simply, a strict textual summary should be shorter than the source, while agreeing semantically. The first condition is straightforward but the second is currently infeasible to ensure with automatic systems, and so we instead enforce extractive summarization to ensure the first and encourage the second.

Without a supervised validation set, there is no clear way to select a value for β_1 in Eq 3 and so no way to optimize this directly. Instead, we opt to ensure both terms improve as our method proceeds. Thus, we are not comparing the pruning and relevance terms directly (only ensuring mutual progress), and so we optimize simpler quantities monotonic in the two terms instead: $p(\tilde{s})$ for pruning and $p(y|\tilde{s})$ for relevance.

We perform extractive summarization by iteratively deleting words or phrases, starting with the original sentence. At each elimination step, we only consider candidate deletions which decrease the value of the pruning term, i.e., increase the language model score of the candidate summary. This ensures progress on the pruning term, and also enforces the notion that word deletion should reduce the information content of the summary. The relevance term is optimized through only expanding candidates that have the highest relevance scores at each iteration, and picking the candidate with the highest relevance score as final summary.

Altogether, this gives 3 principles for extractive summarization with IB.

1. Maximize *relevance term* by maximizing $p(s_{next}|\tilde{s})$.
2. Prune information and enforce compression by bounding: $p(\tilde{s}_{i+1}) > p(\tilde{s}_i)$.
3. Enforce strict summarization by extractive word elimination.

3.2 Method

Algorithm 1 *BottleSum*^{Ex} method

Require: sentence s and context s_{next}

1: $C \leftarrow \{s\}$ ▷ set of summary candidates
2: **for** l in $length(s)...1$ **do**
3: $C_l \leftarrow \{s' \in C | len(s') = l\}$
4: sort C_l descending by $p(s_{next}|s')$
5: **for** s' in $C_l[1:k]$ **do**
6: $l' \leftarrow length(s')$
7: **for** j in $1...m$ **do**
8: **for** i in $1...(l'-j)$ **do**
9: $s'' \leftarrow s'[1:i-1] \circ s'[i+j:l']$
10: **if** $p(s'') > p(s')$ **then**
11: $C \leftarrow C + \{s''\}$
12: **return** $\arg\max_{s' \in C} p(s_{next}|s')$

We turn these principles into a concrete method which iteratively produces summaries of decreasing length by deleting consecutive words in candidate summaries (Algorithm 1). The relevance term is optimized in two ways: first, only the top-scoring summaries of each length are used to generate new, shorter summaries (line 5). Second, the final summary is chosen explicitly by this measure (line 12).

In order to satisfy the second condition, each candidate must contain less self-information (i.e., have higher probability) than the candidate that derives it. This ensures that each deletion (line 9) strictly removes information. The third condition, strict extractiveness, is satisfied per definition.

The algorithm has two parameters: m is the max number of consecutive words to delete when producing new summary candidates (line 9), and k is the number of candidates at each length used to generate shorter candidates by deletion (line 5).

4 Abstractive Summarization with Extractive Self-Supervision

Next, we extend the unsupervised summarization of *BottleSum*^{Ex} to abstractive summarization with

BottleSum^{Self}, based on a straightforward technique for self-supervision. Simply, a large corpus of unsupervised summaries is generated with *BottleSum*^{Ex} using a strong language model, then the same language model is tuned to produce summaries from source sentences on that dataset.

The conceptual goal of *BottleSum*^{Self} is to use *BottleSum*^{Ex} as a guide to learn the notion of information relevance as expressed through IB, but in a way that (a) removes the restriction of extractiveness, to produce more natural outputs and (b) learns an explicit compression function not requiring a next sentence for decoding.

4.1 Extractive Dataset

The first step of *BottleSum*^{Self} is to produce a large-scale dataset for self-supervision using the *BottleSum*^{Ex} method set out in §3.2. The only requirement for the input corpus is that next sentences need to be available.

In our experiments, we generate a corpus of 100,000 sentence-summary pairs with *BottleSum*^{Ex}, using the same parameter settings as in section 3. The resulting summaries have an average compression ratio (by character length) of approximately 0.55.

4.2 Abstractive Fine-tuning

The second step of *BottleSum*^{Self} is fine-tuning the language model on its extractive summary dataset. The tuning data is formed by concatenating source sentences with generated summaries, separated by a delimiter and followed by an end token. The model (GPT-2) is fine-tuned with a simple language modeling objective over the full sequence.

As a delimiter, we use `TL;DR:` , following Radford et al. (2019) who found that this induces summarization behavior in GPT-2 even without tuning. We use a tuning procedure closely related to Radford et al. (2018), training for 10 epochs. We take the trained model weights that minimize loss on a held-out set of 7000 extractive summaries.

To generate from this model, we use a standard beam search decoder, keeping the top candidates at each iteration. Unless otherwise specified, assume we use a beam size of 5. We restrict produced summaries to be at least 5 tokens long, and no longer than the source sentence.

5 Experiments

We evaluate our *BottleSum* methods using both automatic metrics and human evaluation. We find our methods dominant over a range of baselines in both categories.

5.1 Setup

We evaluate our methods and baselines using automatic ROUGE metrics (1,2,L) on the DUC-2003 and DUC-2004 datasets (Over et al., 2007), similar to the evaluation used by Baziotis et al. (2019). DUC-2003 and DUC-2004 consist of 624 and 500 sentence-summary pairs respectively. Sentences are taken from newstext, and each summary consists of 4 human-written reference summaries capped at 75 bytes. We recover next-sentences from DUC articles for *BottleSum*^{Ex}.

We also employ human evaluation as a point of comparison between models. This is both to combat known issues with ROUGE metrics (Schluter, 2017) and to experiment beyond limited supervised domains. Studying unsupervised methods allows for comparison over a much wider range of data where training summary pairs are not available, which we take advantage of here by summarizing sentences from the non-anonymized CNN corpus (Hermann et al., 2015; Nallapati et al., 2016; See et al., 2017).

We use Amazon Mechanical Turk (AMT) for human evaluation, summarizing on 100 sentences sampled from a held out set. Evaluation between systems is primarily done as a pairwise comparison between *BottleSum* models and baselines, over 3 attributes: coherence, conciseness, and agreement with the input. AMT workers are then asked to make a final judgement of which summary has higher overall quality. Each comparison is done by 3 different workers. Results are aggregated across workers and examples.

5.2 Models

In both experiments, *BottleSum*^{Ex} is executed as described in section 3.2. In experiments on DUC datasets, next-sentences are recovered from original news sources, while we limit test sentences in the CNN dataset to those with an available next-sentence (this includes over 95% of sentences). We set parameter $k = 1$ (i.e. expand a single candidate at each step) with up to $m = 3$ consecutive words deleted per expansion. GPT-2 (small) is used as the method's pretrained language model,

with no task-specific tuning. To clarify, the only difference between how *BottleSum*^{Ex} runs on the datasets tested here is the input sentences; no data-specific learning is required.

As with *BottleSum*^{Ex}, we use GPT-2 (small) as the base for *BottleSum*^{Self}. To produce source-summary pairs for self supervision, we generate over 100,000 summaries using *BottleSum*^{Ex} with the parameters above, on both the Gigaword sentence dataset (for automatic evaluation) and CNN training set (for human evaluation). *BottleSum*^{Self} is tuned on the respective set for 10 epoch with a procedure similar to Radford et al. (2019). When generating summaries, *BottleSum*^{Self} uses beam-search with beam size of 5, and outputs constrained to be at least 5 tokens long.

We include a related model, Recon^{Ex} as a simple autoencoding baseline comparable in setup to *BottleSum*^{Ex}. Recon^{Ex} follows the procedure of *BottleSum*^{Ex}, but replaces the next-sentence with the source sentence. This aims to take advantage of the tendency of language models to semantically repeat in to substitute the Information Bottleneck objective in *BottleSum*^{Ex} with a reconstruction-inspired loss. While this is not a perfect autoencoder by any means, we include it to probe the role of the next-sentence in the success of *BottleSum*^{Ex}, particularly compared to a reconstructive method. As Recon^{Ex} tends to have a best reconstructive loss by retaining the entire source as its summary, we constrain its length to be as close as possible to the *BottleSum*^{Ex} summary for the same sentence.

As an unsupervised neural baseline, we include SEQ^3 (Baziotis et al., 2019), which is trained with an autoencoding objective paired with a topic loss and language model prior loss. SEQ^3 had the highest comparable unsupervised results on the DUC datasets that we are aware of, which we cite directly. For human evaluation, we retrained the model with released code on the training portion of the CNN corpus.

We use the ABS model of Rush et al. (2015) as a baseline for automatic and human evaluation. For automatic evaluation, this model is the best published supervised result we are aware of on the DUC-2003 dataset, and we include it as a point of reference for the gap between supervised and unsupervised performance. We cite their results directly. For human evaluation, this model demonstrates the performance gap for out-of-domain summarization. Specifically, it requires supervision (unavailable for the CNN dataset), and so we use the model as originally trained on the Gigaword sentence dataset. This constitutes a significant domain-shift from the first-sentences of articles with limited vocabulary to arbitrary article sentences with diverse vocabulary.

Method	R-1	R-2	R-L
Supervised			
ABS	28.18	8.49	23.81
Li et al. (2017)	31.79	10.75	27.48
Unsupervised			
PREFIX	20.91	5.52	18.20
INPUT	22.18	**6.30**	19.33
SEQ^3	22.13	6.18	19.3
Recon^{Ex}	21.97	5.70	18.81
BottleSum^{Ex}	**22.85**	5.71	**19.87**
BottleSum^{Self}	22.30	5.84	19.60

Table 1: Averaged ROUGE on the DUC-2004 dataset

Method	R-1	R-2	R-L
Supervised			
ABS	28.48	8.91	23.97
Unsupervised			
PREFIX	21.14	**6.35**	18.74
INPUT	20.83	6.15	18.44
SEQ^3	20.90	6.08	18.55
Recon^{Ex}	21.11	5.77	18.33
BottleSum^{Ex}	**21.80**	5.63	**19.19**
BottleSum^{Self}	21.54	5.93	18.96

Table 2: Averaged ROUGE on the DUC-2003 dataset

We include the result of Li et al. (2017) on DUC-2004, who achieved the best supervised performance we are aware of. This is intended as a point of reference for supervised performance.

Finally, for automatic metrics we include common baseline PREFIX, the first 75 bytes of the source sentence. To take into account lack of strict length constraints and possible bias of ROUGE towards longer sequences, we include INPUT, the full input sentence. Because our model is extractive, we know its outputs will be no longer than the input, but may exceed the length of other methods/baselines.

5.3 Results

In automatic evaluation, we find *BottleSum*^{Ex} achieves the highest R-1 and R-L scores for unsupervised summarization on both datasets. This is promising in terms of the effectiveness of the Information Bottleneck (IB) as a framework.

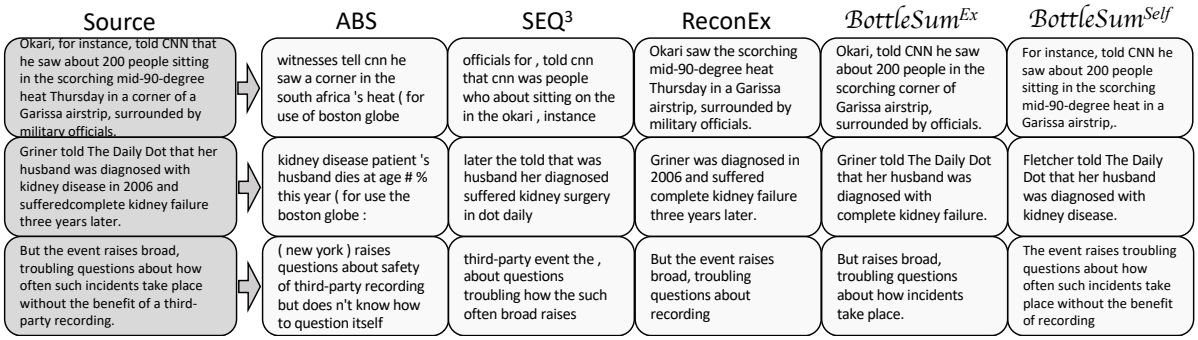

Figure 2: Representative example generations from the summarization systems compared

Models		**Attributes**			**Overall**		
Model	Comparison	coherence	conciseness	agreement	better	equal	worse
BottleSum^{Ex} vs.	ABS	+0.45	+0.48	+0.52	60%	31%	9%
	SEQ^3	+0.61	+0.57	+0.56	61%	34%	5%
	Recon^{Ex}	-0.05	+0.01	-0.05	37%	22%	41%
BottleSum^{Self} vs.	ABS	+0.47	+0.39	+0.48	62%	26%	12%
	SEQ^3	+0.56	+0.45	+0.53	65%	26%	9%
	Recon^{Ex}	+0.11	-0.05	+0.09	47%	14%	39%
	BottleSum^{Ex}	+0.14	+0.06	+0.11	43 %	27%	30%

Table 3: Human evaluation on 100 CNN test sentences (pairwise comparison of model outputs). Attribute scores are averaged over a scale of 1 (better), 0 (equal) and -1 (worse). We also report the overall preferences as percentages.

BottleSum^{Self} achieves the second highest scores in both of these categories, further suggesting that the tuning process used here is able to capture some of this benefit. The superiority of *BottleSum*^{Ex} suggests possible benefit to having access to a relevance variable (next-sentence) to the effectiveness of IB on these datasets.

The R-2 scores for *BottleSum*^{Ex} on both benchmark sets were lower than baselines, possibly due to a lack of fluency in the outputs of the extractive approach used. PREFIX and INPUT both copy human text directly and so should be highly fluent, while Rush et al. (2015) and Baziotis et al. (2019) have the benefit of abstractive summarization, which is less restrictive in word order. Further, the fact that *BottleSum*^{Self} is abstractive and surpasses R-2 scores of both new extractive methods tested here (*BottleSum*^{Ex}, Recon^{Ex}) supports this idea. Recon^{Ex}, also extractive, has similar R-2 scores to *BottleSum*^{Ex}.

The performance of Recon^{Ex}, our simple reconstructive baseline, is mixed. It does succeed to some extent (e.g. surpassing R-1 for all other baselines but PREFIX on DUC-2003) but not as consistently as either *BottleSum* method. This suggests that while some benefit may come from the extractive process of *BottleSum*^{Ex} alone (which Recon^{Ex} shares), there is significant benefit to using a strong relevance variable (specifically in contrast to a reconstructive loss).

Next, we consider model results on human evaluation. *BottleSum*^{Self} and *BottleSum*^{Ex} both show reliably stronger performance compared to models from related work (ABS and SEQ^3 in Table 3). While *BottleSum*^{Self} seems superior to Recon^{Ex} other than in conciseness (in accordance with their compression ratios in Table 4), *BottleSum*^{Ex} appears roughly comparable to Recon^{Ex} and slightly inferior to *BottleSum*^{Self}.

The inversion of dominance between *BottleSum*^{Ex} and *BottleSum*^{Self} on automatic and human evaluation may cast light on competing advantages. *BottleSum*^{Ex} captures reference summaries more effectively, while *BottleSum*^{Self}, through a combination of abstractivness and learning a cohesive underlying mechanism of summarization, writes more favorable summaries for a human audience. Further analysis and accounting for known limitations of ROUGE metrics may clarify these competing advantages.

In comparing these models, there are also practical considerations (summarized in table 5). ABS

can be quite effective, but requires learning on a large supervised training set (as demonstrated by its poor out-of-domain performance in Table 3). While SEQ^3 is unsupervised, it still needs extensive training on a large corpus of in-domain text. *BottleSum*^{Ex}, whose outputs were preferred over both by humans, requires neither of these. Given a strong pretrained language model (GPT-2 small is used here) it only requires a source and next-sentence to summarize. *BottleSum*^{Self} requires in-domain text for self-supervision, but its superior performance by human evaluation and summarization without next-sentence are clear advantages. Further, its beam-search decoding is more computationally efficient than *BottleSum*^{Ex}, which requires evaluating conditional next-sentence perplexity over a large grid of extractive summary candidates.

Another difference from *BottleSum*^{Ex} is the ability of *BottleSum*^{Self} to be abstractivene (Table 4). Other baselines have a higher degree of abstractiveness than *BottleSum*^{Self}, but this can be misleading. Consider the examples in figure 2. While many of the phrases introduced by other models are technically abstractive, they are often off-topic and confusing.

This hints at an advantage of *BottleSum* methods. In only requiring the base model to be a (tunable) language model, they are architecture-agnostic and can incorporate as powerful a language model as is available. Here, incorporating GPT-2 (small) carries benefits like strong pretrained weights and robust vocabulary handling by byte pair encoding, allowing them to process the diverse language of the non-anonymized CNN corpus with ease. The specific benefits of GPT-2 are less central, however; any such language model could be used for *BottleSum*^{Ex} immediately, and *BottleSum*^{Self} with some tuning. This is in contrast architecture-specific models like ABS and SEQ^3, which would require significant restructuring to fully incorporate a new model.

As a first work to study the Information Bottleneck principle for unsupervised summarization, our results suggest this is a promising direction for the field. It yielded two methods with unique performance benefits (Table 1, 2, 3) and practical advantages (table 5). We believe this concept warrants further exploration in future work.

Model	Abstractive Tokens %	Compression Ratio %
BottleSum^{Ex}	-	51
Recon^{Ex}	-	52
BottleSum^{Self}	5.8	56
SEQ^3	12.6	58
ABS	60.4	64

Table 4: Abstractiveness and compression of CNN summaries. Abstractiveness is omitted for strictly extractive approaches

6 Related Work

6.1 Sentence Compression and Summarization

Rush et al. (2015) first proposed abstractive sentence compression with neural sequence to sequence models, trained on a large corpus of headlines with the first sentences of articles as supervision. This followed early work on approaching headline generation as statistical machine translation (Banko et al., 2000). Subsequently, recurrent neural networks with pointer-generator decoders became standard for this task, and focus shifted to the document-level (Nallapati et al., 2016; See et al., 2017).

Pointer-based neural models have also been proposed for extractive summarization (Cheng and Lapata, 2016). The main limitations of this approach are that the training data is constructed heuristically, covering a specific type of sentence summarization (headline generation). Thus, these supervised models do not generalize well to other kinds of sentence summarization or domains. In contrast, our method is applicable to any domain for which example inputs are available in context.

6.2 Unsupervised Summarization

Miao and Blunsom (2016) framed sentence compression as an autoencoder problem, where the compressed sentence is a latent variable from which the input sentence is reconstructed. They proposed extractive and pointer-generator models, regularizing the autoencoder with a language model to encourage compression and optimizing with the REINFORCE algorithm. While their extractive model does not require supervision, results are only reported for semi-supervised training, using less supervised data than purely supervised training. Fevry and Phang (2018) applied denoising autoencoders to fully unsupervised summarization, while Wang and Lee (2018)

Model Name	Model architecture	Data for training	Data for summarizing	When to use
ABS (Rush et al., 2015)	Seq2Seq	Large scale, paired source-summaries (supervised)	source sentence	Large scale supervised training set available
SEQ^3 (Baziotis et al., 2019)	Seq2Seq2Seq	Large scale, unsupervised source sentences	source sentence	Large scale unsupervised (no summaries) data available, *without* next-sentences
BottleSum^{Ex}	Pre-trained LMs	No training data needed	source sentence and next-sentence	No training data available, and next-sentences *are* available for sentences to summarize.
BottleSum^{Self}	Pre-trained LMs (fined-tuned on data from *BottleSum*^{Ex})	Large scale, unsupervised source sentences with next sentences	source sentence	Large scale unsupervised (no summaries) data available, *with* next-sentences and/or no next-sentences available for sentences to summarize

Table 5: Comparison of sentence summarization methods.

proposed an autoencoder with a discriminator for distinguising well-formed and ill-formed compressions in a Generative Adversarial Network (GAN) setting, instead of using a language model. However, their discriminator was trained using unpaired summaries, so while they beat purely unsupervised approaches like ours their results are not directly comparable. Recently Baziotis et al. (2019) proposed a differentiable autoencoder using a gumbel-softmax to represent the distribution over summaries. The model is trained with a straight-through estimator as an alternative to reinforcement learning, obtaining better results on unsupervised summarization. All of these approaches have in common autoencoder-based training, which we argue does not naturally capture information relevance for summarization.

Recently, Zhou and Rush (2019) introduced a promising method for summarization using contextual matching with pretrained language models. While contextual matching requires pretrained language models to generate contextual vectors, *BottleSum* methods do not have specific architectural constraints. Also, like Wang and Lee (2018) it trains with unpaired summaries and so is not directly comparable to us.

6.3 Mutual Information for Unsupervised Learning

We take inspiration from an exciting direction leveraging mutual information for unsupervised learning. Recent work in this area has seen success in natural language tasks (McAllester, 2018; van den Oord et al., 2018), as well as computer vision (Bachman et al., 2019; Hjelm et al., 2019) by finding novel ways to measure and optimize mutual information. Within this context, our work is a further example suggesting mutual information is an important element stimulating progress in unsupervised learning and modelling.

7 Conclusion

We have presented *BottleSum*^{Ex}, an unsupervised extractived approach to sentence summarization, and extended this to *BottleSum*^{Self}, a self-supervised abstractive approach. *BottleSum*^{Ex}, which can be applied without any training, achieves competitive performance on automatic and human evaluations, compared to unsupervised baselines. *BottleSum*^{Self}, trained on a new domain, obtains stronger performance by human evaluation than unsupervised baselines as well as *BottleSum*^{Ex}. Our results show that the Information Bottleneck principle, by encoding a more appropriate notion of relevance than autoencoders, offers a promising direction for progress on unsupervised summarization.

8 Acknowledgments

We thank anonymous reviewers for many helpful comments. This research is supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) (funding reference number 401233309), NSF (IIS-1524371), DARPA CwC through ARO (W911NF15-1-0543), Darpa MCS program N66001-19-2-4031 through NIWC Pacific (N66001-19-2-4031), Samsung AI Research, and Allen Institute for AI.

References

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2017. Unsupervised neural machine translation. *CoRR*, abs/1710.11041.

Philip Bachman, R. Devon Hjelm, and William Buchwalter. 2019. Learning representations by maximizing mutual information across views. *ArXiv*, abs/1906.00910.

Michele Banko, Vibhu O. Mittal, and Michael J. Witbrock. 2000. Headline generation based on statistical translation. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 318–325, Hong Kong. Association for Computational Linguistics.

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 *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL:HLT)*, Minneapolis, USA. To appear.

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

Thibault Fevry and Jason Phang. 2018. Unsupervised sentence compression using denoising auto-encoders. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 413–422, Brussels, Belgium. Association for Computational Linguistics.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in neural information processing systems*, pages 1693–1701.

Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. 2019. Learning deep representations by mutual information estimation and maximization. In *ICLR 2019*. ICLR.

Yann LeCun. 2018. Self-supervised learning: could machines learn like humans? `https://www.youtube.com/watch?v=7I0Qt7GALVk`.

Piji Li, Wai Lam, Lidong Bing, and Zihao Wang. 2017. Deep recurrent generative decoder for abstractive text summarization. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2091–2100, Copenhagen, Denmark. Association for Computational Linguistics.

David McAllester. 2018. Information theoretic co-training.

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

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

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *ArXiv*, abs/1807.03748.

Paul Over, Hoa Dang, and Donna Harman. 2007. Duc in context. *Information Processing & Management*, 43(6):1506–1520.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. Unpublished manuscript.

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

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

Natalie Schluter. 2017. The limits of automatic summarisation according to rouge. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 41–45.

Jurgen Schmidhuber. 1990. Making the world differentiable: On using self-supervised fully recurrent neural networks for dynamic reinforcement learning and planning in non-stationary environments (tr fki-126-90).

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1073–1083.

Naftali Tishby, Fernando C. Pereira, and William Bialek. 1999. The information bottleneck method. In *Proc. of the 37-th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377.

Yaushian Wang and Hung-yi Lee. 2018. Learning to encode text as human-readable summaries using generative adversarial networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4187–4195, Brussels, Belgium. Association for Computational Linguistics.

Jiawei Zhou and Alexander M Rush. 2019. Simple unsupervised summarization by contextual matching. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 5101–5106.

9 Appendix

A Derivation of *BottleSum*^{Ex} Loss

This is a derivation of the loss equation (3) used in section 2, starting with the Information Bottleneck (IB) loss given in eq 2:

$$I(\tilde{S};S) - \beta I(\tilde{S};Y), \tag{4}$$

The goal here is to consider how to interpret this equation when we only have access to single values of source S and relevance variabel Y at a time. In the original IB formulation, a distribution $p(\tilde{S}|S)$ to go from sources to summaries is learned by optimizing this expression across the distribution of source-target pairs (s, y). In the case of *BottleSum*^{Ex}, the goal is to consider this expression on a case-by-case basis, not requiring training over a large distribution of pairs.

First, we consider an alternate form of the equation above:

$$\begin{aligned} &I(\tilde{S};S) - \beta I(\tilde{S};Y) \\ &= \mathop{\mathbb{E}}_{S,\tilde{S}}[pmi(\tilde{S};S)] - \beta \mathop{\mathbb{E}}_{Y,\tilde{S}}[pmi(\tilde{S};Y)] \end{aligned} \tag{5}$$

where $pmi(x,y) = \frac{p(x,y)}{p(x)p(y)}$ denotes pointwise mutual information.

$$= \mathop{\mathbb{E}}_{S,\tilde{S}}\Big[pmi(\tilde{S};S)\Big] - \beta \mathop{\mathbb{E}}_{Y,\tilde{S}}\Big[pmi(\tilde{S};Y)\Big] \tag{6}$$

As stated above, we want to consider this for only single values of s and y at a time, so for these values we can investigate the applicable terms of these expectations:

$$= \sum_{\tilde{s}} \Big[p(s,\tilde{s})pmi(\tilde{S};S) - \beta p(y,\tilde{s})pmi(\tilde{S};Y)\Big] \tag{7}$$

This is the expression we are then aiming to optimize, as it covers all terms in the original IB objective that we have access to on a case-by-case basis.

As in the original IB problem, we can think of learning a distribution $p(\tilde{s}|s)$. However, we are now only taking an expectation over $\tilde{S}$ and so we simply collapse all probability onto the setting of $\tilde{s}$ that optimizes this expression. Simply:

$$p(\tilde{s}|s) = 1 \text{ for chosen summary, } 0 \text{ otherwise} \tag{8}$$

This results in finding $\tilde{s}$ that optimizes:

$$\begin{aligned} &p(s,\tilde{s})pmi(\tilde{S};S) - \beta p(y,\tilde{s})pmi(\tilde{S};Y) \\ =&p(s,\tilde{s})\log\frac{p(s,\tilde{s})}{p(s)p(\tilde{s})} - \beta p(y,\tilde{s})\log\frac{p(y,\tilde{s})}{p(y)p(\tilde{s})} \end{aligned} \tag{9}$$

Any terms that rely only on s and y will be constant and so can be collected into coefficients. As well, remembver that we set $p(\tilde{s}|s) = 1$. Doing rearranging:

$$\begin{aligned} =&p(\tilde{s}|s)p(s)\log\frac{p(\tilde{s}|s)}{p(\tilde{s})} - \beta p(y|\tilde{s})p(\tilde{s})\log\frac{p(y|\tilde{s})}{p(y)} \\ =&c_1\log\frac{1}{p(\tilde{s})} - \beta p(y|\tilde{s})p(\tilde{s})\log p(y|\tilde{s}) - c_2 \end{aligned} \tag{10}$$

This is equivalent to optimizing:

$$\log\frac{1}{p(\tilde{s})} - \beta_1 p(y|\tilde{s})p(\tilde{s})\log p(y|\tilde{s}) \tag{11}$$

for some positively signed β_1.