EASE: Extractive-Abstractive Summarization End-to-End using the Information Bottleneck Principle

Anonymous EMNLP submission

Abstract

Current abstractive summarization systems outperform their extractive counterparts, but their widespread adoption is inhibited by the inherent lack of interpretability. Extractive summarization systems, though interpretable, suffer from redundancy and possible lack of coherence. To achieve the best of both worlds, we propose EASE, an extractive-abstractive framework that generates concise abstractive summaries that can be traced back to an extractive summary. Our framework can be applied to any evidence-based text generation problem and can accommodate various pretrained models in its simple architecture. We use the Information Bottleneck principle to jointly train the extraction and abstraction in an end-to-end fashion. Inspired by previous research that humans use a two-stage framework to summarize long documents (Jing and McKeown, 2000), our framework first extracts a pre-defined amount of evidence spans and then generates a summary using only the evidence. Using automatic and human evaluations, we show that the generated summaries are better than strong extractive and extractive-abstractive baselines.

1 Introduction

Pretrained sequence-to-sequence language models such as BART (Lewis et al., 2020), T5 (Raffel et al., 2019) and their variants have achieved state-of-the-art results on various tasks such as summarization, machine translation, and data2text tasks (Zhang et al., 2019b; Kale and Rastogi, 2020). Despite the higher fidelity compared with models without pretraining for tasks such as summarization (Maynez et al., 2020), the lack of interpretability in abstractive generation remains an obstacle to their broader adoption. Extractive summarization systems, on the other hand, have the advantage of being interpretable but are too restrictive by forcing the output to be spans from the document, reducing their naturalness, coherence, and conciseness. In this paper, we propose EASE, a novel framework that combines the two systems to produce natural summaries that can be traced back to an interpretable extractive summary. Our general framework can accommodate differnet pretrained models and suitable for any evidence-based text generation task.

Source Document:
(CNN)Mike Rowe is coming to a river near you. "Sometimes, you hear about a person who makes you feel good about humanity, but bad about yourself," Rowe says. **On Thursday's episode of "Somebody's Gotta Do It," Rowe meets up with Chad Pregracke, the founder of Living Lands & Waters, who does just that.** Pregracke wants to clean up the nation's rivers one piece of detritus at a time. His quota? Always "more." Read Mike Rowe's Facebook post on how to break our litter habit. **Since he founded the nonprofit in 1998 at the ripe age of 23, Pregracke and more than 87,000 volunteers have collected 8.4 million pounds of trash from U.S. waterways. Those efforts helped him earn the 2013 CNN Hero of the Year Award, along with numerous other honors.** "Wherever you are, no matter if there's a stream, a creek, a lake, whatever, that needs to be cleaned up, you can do it. Just organize it and do it," he told CNN's Anderson Cooper after his win. Pregracke also gives Rowe a tour of the 150-foot, solar-powered barge that the Living Lands & Waters staff calls home during lengthy cleanups. The part-home, part-office, part-dumpster has seven bedrooms, two bathrooms, a classroom and a kitchen – and just happens to be made from a recycled strip club. According to the organization's latest annual report, Pregracke has made it his mission in 2015 to remove 500,000 more pounds of trash. If you'd like to help achieve this goal, visit his website to learn how to help: LivingLandsAndWaters.org/Get-Involved/.

Summary: Mike Rowe meets Chad Pregracke, the founder of Living Lands & Waters. The nonprofit has collected 8.4 million pounds of trash from U.S. waterways. Pregracke was named the 2013 CNN Hero of the Year.

Figure 1: An example of a summary and its evidence (highlighted) as generated by our framework.

The existing extractive-abstractive systems can be divided into three main categories: 1- Relying on attention for interpretability (Hsu et al., 2018). Due to the probabilistic nature of the attention mechanism, it falls short of providing usable evidence; 2- Providing word-level evidence for the generated summaries (Gehrmann et al., 2018). Though more useful than attention, this evidence is too granular to be useful for humans; 3- Training the content selector separately using pseudo labels or other heuristics (Liu and Lapata, 2019; Pilault et al., 2020). In contrast, we seek a theoretically-grounded model that can learn the evidence extraction end-to-end.

Perhaps the closest work to ours is Zhao et al. (2020) focusing on long-document summarization by training a joint extractive-abstractive model via weak supervision. Though a complicated and spe-

cific framework, it achieves poor results on benchmarks such as CNN/DM. EASE on the other hand, is based on the Information Bottleneck (IB) principle (Tishby et al., 1999), which formalizes the trade-off between the size of the extracted evidence and the information provided for the generation of the final output. While this method has been successfully adopted by prior work for a simpler discriminative task (Paranjape et al., 2020), we extend it to generative tasks where the extracted evidence can be viewed as a coarse version of the final abstractive output.

We leverage pretrained language models that first extract the necessary evidence from the source document (*extractor*) and then, using only the extracted evidence spans, generate the final output (*abstractor*). Fig. 1 shows an example of the evidences and summary generated by our system.

Our main contributions are as follows:

- We propose EASE, a general-purpose theoretically-grounded Extractive-Abstractive framework for extractive-abstractive text generation that is jointly trained in an end-to-end fashion. We apply EASE to text summarization.
- Our abstractor generates the summary using only the extracted evidence which can be viewed as an extractive summary. We propose a new *sparsity budget* parameter that controls the trade-off between the length of the evidence spans (i.e., the extractive summary)and the final abstractive output's quality
- Our results show that EASE extracts evidence better than the baselines without significantly sacrificing the quality of the generated summary, compared with the state-of-the-art fully abstractive systems on the CNN/DailyMail dataset.

2 Extractive-Abstractive Framework

There exists evidence that humans use a two-stage extractive-abstractive framework to summarize long documents (Jing and McKeown, 2000) by first extracting salient parts and then deciding what to eliminate, reword, and reorganize. Inspired by this, we propose EASE, a framework that learns extraction and abstraction collectively in an end-to-end fashion. This not only provides interpretable evidence for the generated summary, which can be many times smaller than the original document, but also reduces the effective input length used during abstraction. This has been shown to directly correlate with the extent of hallucination in pretrained language models (Yang et al., 2020a).

In order to formalize the problem, we use the IB principle to learn an optimal model between the original document x and the final summary y through a compressed representation z. The IB objective is to minimize the following:

$$L_{IB} = I(x; z) - \beta I(z; y), \quad (1)$$

where $I()$ is the mutual information. This objective encourages z to contain only the information about x that is useful in predicting y. Moreover, β controls the trade-off in z between containing information about x (i.e., sparsity) vs about y (i.e., prediction quality).

We use a relaxation for (1) similar to Paranjape et al. (2020) to make it tractable. As such, z is obtained by masking the original document x to produce a summaries y. We illustrate EASE in Fig. 2. EASE can perform extraction (i.e., masking) either at the token or at the sentence level. We first describe the token-level model and subsequently generalize it for sentence-level extraction. As such, the extractor masks tokens in the original document x to extract a rough summary z, which is used as evidence by the abstractor to produce the summary y. We define $z = m \odot x$ where m is a boolean mask on the input x. This is similar to the masking process used in Masked Language Models (MLM), except that instead of random masking (Devlin et al., 2019) or heuristic-based masking (Zhang et al., 2019b,d), we learn which tokens should be masked in an end-to-end fashion. Using the variational bound (Alemi et al., 2016) on (1), the model is trained using two loss terms. The first loss ensures that the final summary is close to the golden summaries:

$$L_{Task} = E_{m \simeq p(m|x)}[-\log q_\theta(y|m \odot x)], \quad (2)$$

where $q_\theta(y|z)$ is a parametric approximation to the true likelihood $p(y|z)$.

Similar to Paranjape et al. (2020), we assume that the mask variables over individual words are conditionally independent given the input x. This means that the evidence z can contain redundancies, as the extractor chooses evidence individually without conditioning on prior extractions. Since the extracted evidence is not the final summary,

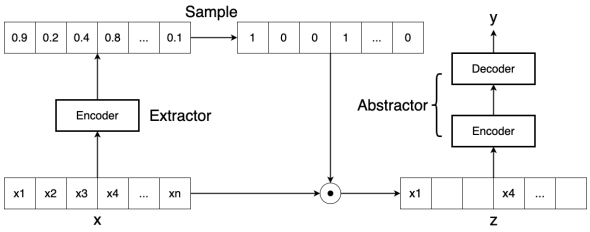

Figure 2: The Extractive-Abstractive model architecture. The extractor samples the evidence from the source which is used by the abstractor.

the abstractor still has the opportunity to eliminate redundancies. Nallapati et al. (2017) explore a modeling approach that keeps track of the current state of the summary, but we leave this direction to future work. Formally,

$$p_\theta(z|x) = \prod_j p_\theta(z_j|x),$$

where $p_\theta(z_j|x) = Bernoulli(\theta_j(x))$.

Optimizing the loss in (2) would result in the extractor masking no tokens and hence, maximizing the mutual information between the input and output of the abstractor. Therefore, the second loss term is a sparsity constraint to ensure that the extractor's output is a measurable subset of input tokens and can be used as evidence for the abstractor output:

$$L_{Sparsity} = \sum_j KL[p_\theta(z_j|x), r(z_j)], \quad (3)$$

where we set the prior distribution $r(z_j) = Bernouli(\pi)$. For summarization tasks π can be small i.e. $0.3 \leq \pi \leq 0.5$. As such, the combined loss can be written as:

$$\begin{aligned} L_{EA} =& E_{m\simeq p(z|x)}[-\log q_\theta(y|m \odot x)] \\ &+\beta \sum_j KL[p_\theta(z_j|x), Bernouli(\pi)], \end{aligned} \quad (4)$$

where $p_\theta(z|x)$ is the parametric posterior distribution over z and β is a hyperparameter to weigh the performance-sparsity trade-off.

2.1 Soft Masking

The combined loss presented above is not differentiable, as it includes sampling operations from Bernoulli distributions. Since we aim to learn the masking function (unlike random masking), this would not be amenable to end-to-end training using backpropagation. Rather than using the REINFORCE algorithm which suffers from high variance (Bastings et al., 2019), we use the Gumbel Softmax reparameterization trick (Jang et al., 2017) similar to Paranjape et al. (2020). This replaces the sampling step with an argmax: $argmax_{i\in 0,1}(logp(z_j|x) + g_i)$, where g_i is a random sample from the Gumbel$(0, 1)$ distribution. Finally, the argmax is replaced by a weighted softmax:

$$z_j^* = \frac{\exp\left((\log(p(z_j = 1|x) + g_1)/\tau\right)}{\sum_{i\in 0,1} \exp\left((\log(p(z_j = i|x) + g_i)/\tau\right)}.$$

Note that $z_j^* \in (0, 1)$ gets boundary values (i.e., 0 or 1) when $\tau \to 0$ (in practice, we use $\tau = 0.01$).

2.2 Model Architecture

As illustrated in Fig. 2, our model has two parts: the extractor and the abstractor. The extractor is a pretrained transformer encoder similar to BERT (Devlin et al., 2019) with an additional linear layer on top that computes $p_\theta(z_j|x)$. The abstractor on the other hand, is a pretrained seq-to-seq language model like BART (Lewis et al., 2020). From our experiments, we find a BART-base encoder (6 layers) to be adequate as an extractor model, while we use a BART-large abstractor. Note that we can use any other pretrained encoders (e.g., RoBERTa (Liu et al., 2019)) and seq2seq models (e.g., Pegasus (Zhang et al., 2019b)) for the extraction and abstraction task, respectively. Also note that after the evidence extraction, in order to ensure that there is no leakage of information, we need to encode the extracted tokens separately again. Using the same encoded representation would leak information to the abstractor about the masked tokens.

During training, given an input x, the extractor generates a probability for each token in x to be selected (i.e. not masked). Based on these probabilities (p_j), we soft-sample m_j with values in (0,1). We then pass $z = x \odot m$ to the abstractor to generate the output. In our experiments, we tried two different ways of masking the input using m: 1) directly masking the embedding, i.e. $z_j = m_j * x_j + (1 - m_j) * x_{mask}$ where x_{mask} is initialized from the BART's original `<mask>` token, and, 2) using m as an attention mask for both the encoder's self attention as well as the encoder-decoder cross attention, i.e. to block attention to the masked tokens. However, we did not observe a

significant difference between these two schemes. During the inference, the extractor deterministically selects the top $\pi\%$ of the source tokens. Such hard masking ensures that the sparsity requirement is exactly met during inference time.

2.3 Sentence-level Extraction

In the previous section, we described token-level extraction where each token in the source document is individually masked or retained. The main drawback of using scattered token-level extraction is that it is difficult to be used as interpretable evidence. While in Section 5.1, we explore a method for improving the interpretability of token-level evidence by encouraging span-level extraction, in this section, we focus on sentence-level extraction as an effective means to achieve interpretability.

In sentence-level extraction approaches, the model first selects the sentences that need to be masked, followed by the masking of all tokens within those sentences. Unlike the token-level model, the extractor's output in this setup is a linguistically plausible (but possibly redundant) extractive summary, i.e., complete sentences from the source. For sentence-level extraction, we add a special `[CLS]` token to the beginning of each sentence and use its representation as the sentence encoding. We also add a segment embedding to each token in the sentence to distinguish between the sentences in a document. The segment embeddings are initialized randomly and learned during training. We use the `[CLS]` token representation to perform soft masking as in the token-level model.

3 Experimental Settings

Datasets: We primarily experiment with the CNN/DailyMail dataset (Hermann et al., 2015) owing to its extractive-like nature; its summaries are typically closely related to the source sentences. We also present results on the XSUM (Narayan et al., 2018) dataset, a highly abstractive dataset in which summaries can be viewed as a title for the source documents.

Model Hyperparameters and evaluation metrics: We initialize the seq-to-seq abstractor with the BART-large model and initialize the extractor with the BART-base encoder.
We use the fairseq codebase[1] for our experiments and use the same hyperparameters as used for fine-tuning BART on CNN/DM and XSum by the official codebase. Specifically, we fine-tune BART using a polynomial decay learning rate scheduler with the Adam optimizer (Kingma and Ba, 2014). We use a learning rate of 3e-5 with 500 warmup steps and train for 20000 steps. During our initial experiments, we observed similar results for values of $\beta \in [1, 10]$ in (4). We use $\beta = 5$ in our reported results. We use ROUGE F1 scores (R1/R2/RL) for the automatic evaluation. ROUGE scores were calculated using the files2rouge toolkit[2].

4 Results

In this section, we report the performance of our model from both automatic and human evaluation perspective, along with ablation studies. Figure 4 shows example summaries along with evidence highlighted from our system at different sparsity levels.

4.1 Automatic Evaluation

In Table 1, we present the performance of our model for CNN/DM and XSum when using a sparsity of 0.5, with a BART-base encoder as the extractor and a BART-large abstractor. We also present the performance of BART and BERTSUM as representative abstractive and extractive systems, respectively. Moreover, they can be considered as EASE's exctractor (BERTSUM) or abstractor (BART) on their own. Note that for BERTSUM, we present the performance of the Ext-large version for CNN/DM and the two-stage ExtAbs version for XSum. We also include results from previous evidence-based extractive-abstractive systems for comparison. For CNN/DM, our token-level and sentence-level models that use around 50% of the source input perform slightly better than BERTSUM, but slightly worse than BART-large. For XSum, our gap with the BART-large baseline is larger. This is expected given that XSum summaries are highly abstractive, making it much harder for the extractor to extract the most important information in an end-to-end fashion.

Moreover, we observe that the sentence-level model performs slightly better than the token-level model for CNN/DM but slightly worse for XSum. We hypothesize that for the more extractive CNN/DM dataset, keeping continuous spans of text is of paramount importance, while for the more abstractive XSum dataset, the sparsity budget

[1] https://github.com/pytorch/fairseq

[2] https://github.com/pltrdy/files2rouge

Model	CNN/DailyMail	XSum
BART-large (Lewis et al., 2019)	44.16/21.28/40.90	45.14/22.27/37.25
BERTSUM (Liu and Lapata, 2019)	43.85/20.34/39.90	38.81/16.50/31.27
Previous evidence-based Extractive-Abstractive systems		
Bottom-Up (Gehrmann et al., 2018)	40.96/18.38/38.16	-
SEAL (Zhao et al., 2020)	39.3/16.5/-	-
EASE (ours)		
Token-level sparsity 0.5	43.96/20.91/40.74	42.70/19.38/33.81
Sentence-level sparsity 0.5	43.98/20.95/40.78	41.82/19.05/33.99

Table 1: ROUGE-1/2/L results for CNN/DailyMail and XSum.

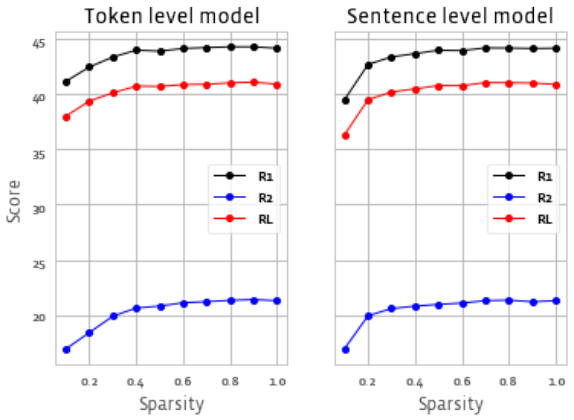

Figure 3: R1/R2/RL vs Sparsity for token level and sentence level models. For sentence level model, we enforce it to extract at least three sentences.

Model	Token level	Sentence Level
base Ex + base Ab	42.9/19.8/39.7	42.42/19.7/39.27
base Ex + base Ab shared	42.28/19.37/39.14	42.58/19.81/39.43
base Ex + large Ab	43.96/20.91/40.74	43.98/20.95/40.78

Table 2: Ablation studies on the effect of model size and sharing. All models are trained with 0.5 sparsity.

can be better spent on a more scattered extraction of key pieces throughout the document. In section 5, we explore ideas to 1) improve the performance of the token-level model using pre-training; 2) improve the interpretability of token-level models by encouraging the extraction of continuous spans; and 3) improve the performance of both token and sentence level models using semi-supervised learning.

4.2 Model Analysis

Effect of Sparsity Prior: In this section, we investigate the effect of sparsity on the generated summaries. Figure 3 presents ROUGE score of both token-level and sentence-level models, trained with different sparsity priors. As expected, increasing the sparsity ratio improves the ROUGE scores at the cost of more verbose extracted evidence. Moreover, the performance gains flatten after a sparsity of around 0.3. We found that token-level models are more robust to lower sparsity rates, i.e. they can remove functional words without losing document information, but they are not well-suited in terms of interpretability. Note that for the sentence-level models, at inference time we extracted at least three sentences to ensure that short documents would have enough evidence at lower sparsity rates.

Effect of model size: We examine the effect of using models of different sizes on summarization performance, and also explore the possibility of sharing the encoder. We consider BART-base and BART-large for the abstractor. We also experimented with using RoBERTa (Liu et al., 2019) and BART-large encoder for the extractor but found it very unstable and hard to tune the relative loss weights. To explore the possibility of reducing the model size, we also experiment with sharing the encoder's parameters between the extractor and abstractor encoders. Table 2 presents results of these settings for both token-level and sentence-level models using a sparsity of 0.5.

We can see that using a large model for the abstractor yields significant improvements. Moreover, sharing the encoder between the extractor and the abstractor does not hurt the performance. However, since using a large abstractor is essential while using a large extractor is unstable during training, we use a BART-base extractor and a BART-large abstractor for our default setting.

Effect of extraction: We evaluate the effect of extraction quality on the final summary for our

Extract Sentences	Sparsity 0.5	Sparsity 0.3
$\pi\%$	43.98/20.95/40.78	43.38/20.57/40.2
Top-3	41.37/18.58/38.18	41.19/18.63/38.01
Lead-3	40.84/18.16/37.71	40.6/18.05/37.46
Random-3	31.46/10.04/28.92	31.34/10.06/28.79

Table 3: Effect of different extraction techniques on the final summary.

Models	Summary		Extraction
	Consistency	Relevance	Relevance
BART	**4.89**	**4.13**	-
Token-level model	4.77	**4.16**	-
Sentence-level model	**4.86**	3.80	**4.45**
Lead-3 extraction	-	-	4.38

Table 4: Human Evaluation results on CNN/DM. We evaluate our token-level and sentence-level models, with 0.5 sparsity on summary relevance and consistency and compare with BART. We evaluate extraction relevance of our sentence-level model and compare with Lead-3.

sentence-level models. We use our model trained with different sparsity rates but during inference, feed only the top-3 sentences with highest scores to the abstractor for generating the summary. We compare with the baselines of using random-3 and lead-3 sentences as well as using all $\pi\%$ of sentences. Table 3 presents results of our two models with sparsity values of 0.5 and 0.3. We find that for both models, summaries using the top-3 sentences selected by the extractor outperform lead-3 extraction, even though the CNN/DM dataset has a strong lead bias. We conclude that our extractor is indeed extracting important sentences, which we further confirm using human evaluations, described in the next section.

4.3 Human Evaluation

We conduct human evaluation on both the extracted evidence and the generated summaries. For the summaries, we asked annotators to rate them between 1-5 on two qualitative aspects of the summary: Consistency and Relevance. Consistency is the factual alignment between the summary and the source document, measuring whether the summary is changing details or hallucinating. Relevance measures whether the summary captures the key points of the source document. We compared our generated summaries with BART as a baseline. We also evaluate the relevance of extractions from the sentence-level models. To make evaluation easier, we gather the top-3 sentences with the highest extraction scores and ask annotators whether those are the most important sentences in the source document. Here, we compare with Lead-3 extraction as a baseline.

We sampled 200 examples from the CNN/DM test set and conducted human evaluation using Amazon Mechanical Turk with three annotators. We present the average annotators' scores in Table 4, using z-score p-values smaller than 0.01 to measure statistical significance. We find that for extraction relevance, the top-3 sentences from our extractor scored higher than Lead-3, which itself received a high relevance score due to the strong lead bias in the CNN/DM dataset. For abstractive summaries, we find that the sentence-level model achieves a similar consistency score as BART, but slightly better than the token-level model. On one hand, the sentence model achieves a lower relevance score than BART and token model. We hypothesize that the interpretable nature of the sentence model results in a loss of some of the key information in the source document as expected, whereas the token model avoids this by extracting keywords throughout the source. On the other hand, the token-level model can fabricate new details between the extracted keywords, which results in lower consistency. As such, there is an inherent trade-off between relevance and interpretability.

5 Further improvements and Future Work

5.1 Span-level model with Lasso loss

In the previous section, we found that although sentence-level models are interpretable, they can miss out on key parts of the source document. However, token-level models enjoy much more freedom during extraction but yield evidence that is not very useful for humans. To find a compromise between these two, i.e. a span-level model, we attempt to make the evidence extracted by token-level models more contiguous, by adding a lasso loss (Bastings et al., 2019) to the total loss in (4):

$$L_{Lasso} = \sum_{i=0}^{n-1} |z_i - z_{i+1}|, \qquad (5)$$

where n is the number of source tokens. The lasso loss ensures that the number of transitions between the masked and unmasked tokens is minimized and hence, the model extracts more contiguous spans of text as evidence. In the first row of Table 5, we observe that the lasso loss mainly improves the

Source Document:
(CNN)Two passengers found dead on a cruise ship in Puerto Rico appear to have died in a murder-suicide, the cruise line said. Holland America Line said two guests were found dead inside their stateroom on the ms Ryndam at 11:30 a.m. Thursday. "The cabin was immediately secured, and the authorities were notified, including the FBI," Holland America said. "We are cooperating fully with the investigation, and the authorities will make the official determination on what occurred." **FBI spokesman Moises Quinones said authorities were on scene investigating.** The ship left Tampa, Florida, on March 29 on a 14-day Southern Caribbean cruise. It's currently in San Juan, Puerto Rico. Puerto Rico Port Authority spokesman Efraín Santiago told El Nuevo Dia newspaper that the cleaning staff on the ship had discovered the deceased passengers after knocking on the cabin's door.
Summary (Sparsity 0.3): Holland America Line said two guests were found dead inside their stateroom on the ms Ryndam at 11:30 a.m. Thursday. The FBI is investigating.

Source Document:
(CNN)Gastrointestinal illness has gripped 100 people on the cruise ship Celebrity Infinity, according to a report from the Centers for Disease Control. Of the ship's 2,117 passengers, 95 have suffered from vomiting, diarrhea and other symptoms, the CDC said. The illness has also affected five members of the 964-person crew. The CDC has yet to determine what's causing the ailments. Two staffers from the agency are scheduled to meet the West Coast-based ship in San Diego on Monday. The Infinity left San Diego on March 29. It made its last stop in Puerto Vallarta, Mexico, on April 10, according to MarineTraffic.com. **Celebrity Cruises has been taking action since the outbreak began, including increasing cleaning and disinfection procedures, keeping passengers informed and taking specimens from the afflicted for testing by the CDC, the agency says.** According to the Maritime Executive, this is the third time the Celebrity Infinity has suffered an outbreak of gastrointestinal illness, with others occurring in 2006 and 2013. The ship was built in 2001 and refurbished in 2011.
Summary (Sparsity 0.5): Of the ship's 2,117 passengers, 95 have suffered from vomiting, diarrhea. The illness has also affected five members of the 964-person crew. Celebrity Cruises has been taking action since the outbreak began.

Figure 4: Summarization outputs with their evidence (highlighted), from our systems at different sparsity levels.

token-level model. This is particularly evident in the improvement in RL which is due to the extraction of contiguous spans as evidence.

5.2 Unlabeled Pretraining

Although we initialize the extractor and abstractor with pretrained language models, the model may benefit from further pretraining suited to the downstream task. To this end, we use our model in an auto-encoding fashion, i.e., the abstractor reconstructs the original text using the extracted pieces selected by the extractor. Our hypothesis is that an extractor capable of extracting the most informative parts from which the source can be reconstructed should be better positioned to extract important parts of the source, resulting in higher-quality summaries. Therefore, we pretrain EASE on the WikiText-103 (Merity et al., 2017) dataset to reconstruct the original unlabeled documents using the same loss as in (4) by setting $Y = X$. This can be viewed as a special case of summarization, where the compression rate is one. We only pretrain the token-level model, since pretraining sentence-level models without measures such as topic guidance (Kang and Hovy, 2020) typically leads to hallucination. Results on the CNN/DM dataset by adding pretraining are presented in the second row of Table 5. Even though pretraining improves the token-level model, results for the span-level model are mixed. Our hypothesis is that the lasso continuity helps with summarization by picking contiguous spans, as evidenced by the high RL. However, during the reconstruction pretraining, the lasso loss can be problematic by masking long spans, which are then prone to hallucinations. We leave pretraining alongside span extraction using techniques such as guided reconstruction to future work.

Model	Token level	Span Level (lasso)
vanilla EASE	43.96/20.91/40.74	44.33/20.67/41.06
+ pretraining	44.12/20.89/40.80	44.06/20.82/40.83

Table 5: CNN/DM results on token-level models trained with lasso loss and pretraining.

Model	Token level	Sentence level
vanilla EASE	43.96/20.91/40.74	43.98/20.95/40.78
+ SSL	44.28/21.21/41.0	44.10/21.12/40.89

Table 6: Results on token level and sentence level models, trained with additional semi-supervised extraction.

5.3 Semi-supervised Training

Multiple recent works (Nallapati et al., 2017; Liu and Lapata, 2019) have explored heuristics to obtain pseudo alignments between target summaries and source sentences for summarization datasets. To evaluate the effect of weakly supervising the extractor in EASE using these pseudo labels, we use the greedy procedure of Liu and Lapata (2019) to obtain oracle extractive annotations for CNN/DM. As such, we maintain an evidence set and greedily add source sentences to the set that yield the maximum increase in its ROUGE score against the target summary. This yields a binary labeling of input sentences and we introduce an additional binary cross entropy loss to our training objective in (4) between this binary labeling and the predicted masking probabilities. By using the sentence-level pseudo labels for the tokens of each sentence, we also add this loss to the token-level models. We have shown the results in Table. 6. We observe improvements in all ROUGE metrics for both sentence-level and token-level models, though the gains on the former are more modest. Studying the interaction of this objective with the aforementioned lasso objectives is left for future work.

6 Related Work

6.1 Pretrained Models for Summarization

Lewis et al. (2020) introduced BART, a general-purpose denoising seq2seq transformer, that achieved the state-of-the-art results on many summarization tasks. Later, Zhang et al. (2019b) extended the MLM denoising objective using sentence masking. Zhang et al. (2019c) introduced a multi-stage encoder for extractive summarization, whereas Zhang et al. (2019a) use a two-stage decoder to generate summaries by creating a draft and refining it using a pretrained language model. In EASE, we use pretrained models, i.e., BART, to initialize the extractive and abstractive modules but after that, use an end-to-end loss that trains both modules simultaneously.

6.1.1 Self-supervised Summarization

Miao and Blunsom (2016) introduced an autoencoder setup for sentence compression to reduce the need for labeled examples. A copy ptr/generator model was used for the compressor which alongside the reconstructor is trained to reconstruct the unlabeled documents. Moreover, REINFORCE (Williams, 1992) was used to train the model end-to-end. Baziotis et al. (2019) introduced a similar autoencoder setup but used the Gumbel Softmax reparametrization for training. (Févry and Phang, 2018) also used a denoising autoencoder to compress sentences and a countdown at the decoder to control summary length.

Inspired by the IB principle, West et al. (2019) introduced a recursive algorithm to prune a document to form an unsupervised extractive summary. These summaries are in turn used to train a self-supervised system using a next-sentence objective is used. In contrast, we use a loss formulation derived directly from the IB and train the model end-to-end. (Saito et al., 2020) used a saliency model to extract important pieces of a document before feeding them to an abstractive seq2seq model. In contrast with our model, the saliency module is trained separately by using heuristics to provide pseudo labels for the extraction. (Yang et al., 2020b) proposed pretraining over millions of news articles using the lead sentence as the self supervision.

6.2 Extractive-Abstractive Summarization

The transformer decoder (Liu* et al., 2018) was first used to accommodate long documents from a coarse extractive summarizer. Later, Zhao et al. (2020) also focus on long-document summarization and train a joint extractive-abstractive model by weakly supervising the extractor through pseudo labels. This model, although interpretable, does poorly on a dataset like CNN/DM. (Pilault et al., 2020) introduce another interpretable summarizing model for long documents by performing a simple extractive step to condition the decoder. They show that this approach produces more abstractive summaries compared with the copy mechanism. Unlike these models, we train both modules jointly using the theoretically grounded IB principle with no pseudo labels. Moreover, we seek consistent models suitable for more extractive datasets and achieve results on par with the abstractive model while only using half of the input. (Gehrmann et al., 2018) trained a content selector separately to tag the words and then use bottom-up attention to only copy words from the tagged set. Similar to our token-level model, this is not useful evidence.

Compressive summarization is another way to have a trade-off between extractive and abstractive methods where extractive summaries are compressed to form the final summary (Mendes et al., 2019). Recently, Desai et al. (2020) use syntactic rules to find a high-recall candidate set and then use the notions of plausibility and salience to ensure the grammaticality and importance of the remaining pieces, respectively. Unlike compressive summarization, we explore an extractive-abstractive framework where a concise abstractive summary can be traced back to the evidence; learned jointly with no manual rules or postprocessing.

7 Conclusion

In this paper, we introduced EASE, an extractive-abstractive framework for summarization tasks that trains an extractor and an abstractor in an end-to-end fashion. The extracted evidence can be viewed as an interpretable extractive summary of the summary from which the final summary is generated by the abstractor. We show that our sentence-level extractive-abstractive summarization systems are better than strong extractive-abstractive baselines and either on-par or only slightly lower in quality compared to strong abstractive baselines.

8 Ethical Considerations

Intellectual Properties and Privacy Rights All of the datasets (CNN/DM and XSum) used in our study are publicly available. Regarding privacy rights, the authors of the paper completed IRB human subject protection training for conducting this study.

Compensation for Annotators We compensated the Turkers approximately $15 per hour. We first annotated examples in-house to determine the required annotation speed. We evaluated 200 examples with 8 annotations per example (including outputs from different models) and typically each example takes around 10 minutes.

Steps Taken to Avoid Potential Problems We interacted closely with the Turkers to ensure that compensation was fair and that the instructions were clear. We did pilot examples with each annotator in the beginning to help them to be better calibrated.

Environmental Cost The experiments described in the paper make use of V100 GPUs with 32GB memory. We used up to 8 GPUs per experiment. The experiments may take several hours. We didn't do a lot of parameter search: we re-used the best parameter reported from BART open-source code and only tuned weight on loss on the validation set. Future work will be able to draw on these insights and models in production may be trained once for use using the most promising settings.

References

Alexander Alemi, Ian Fischer, Joshua Dillon, and Kevin Murphy. 2016. Deep variational information bottleneck. *ICLR*.

Jasmijn Bastings, Wilker Aziz, and Ivan Titov. 2019. Interpretable neural predictions with differentiable binary variables. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2963–2977, Florence, Italy. Association for Computational Linguistics.

Christos Baziotis, Ion Androutsopoulos, Ioannis Konstas, and Alexandros Potamianos. 2019. SEQˆ3: Differentiable sequence-to-sequence-to-sequence autoencoder for unsupervised abstractive sentence compression. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 673–681, Minneapolis, Minnesota. Association for Computational Linguistics.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Thibault Févry 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.

Sebastian Gehrmann, Yuntian Deng, and Alexander Rush. 2018. Bottom-up abstractive summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4098–4109, 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*, volume 28, pages 1693–1701. Curran Associates, Inc.

Wan-Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, Jing Tang, and Min Sun. 2018. A unified model for extractive and abstractive summarization using inconsistency loss. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 132–141, Melbourne, Australia. Association for Computational Linguistics.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax.

Hongyan Jing and Kathleen McKeown. 2000. The decomposition of human-written summary sentences.

Mihir Kale and Abhinav Rastogi. 2020. Text-to-text pre-training for data-to-text tasks. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 97–102, Dublin, Ireland. Association for Computational Linguistics.

Dongyeop Kang and Eduard Hovy. 2020. Plan ahead: Self-supervised text planning for paragraph completion task. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6533–6543, Online. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. Cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

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.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Peter J. Liu*, Mohammad Saleh*, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences. In *International Conference on Learning Representations*.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

Afonso Mendes, Shashi Narayan, Sebastião Miranda, Zita Marinho, André F. T. Martins, and Shay B. Cohen. 2019. Jointly extracting and compressing documents with summary state representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3955–3966, Minneapolis, Minnesota. Association for Computational Linguistics.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

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, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. AAAI'17. AAAI Press.

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

Bhargavi Paranjape, Mandar Joshi, John Thickstun, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. An information bottleneck approach for controlling conciseness in rationale extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1938–1952, Online. Association for Computational Linguistics.

Jonathan Pilault, Raymond Li, Sandeep Subramanian, and Chris Pal. 2020. On extractive and abstractive neural document summarization with transformer language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9308–9319, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer.

Itsumi Saito, Kyosuke Nishida, Kosuke Nishida, and Junji Tomita. 2020. Abstractive summarization with combination of pre-trained sequence-to-sequence and saliency models.

Naftali Tishby, Fernando C. Pereira, and William Bialek. 1999. The information bottleneck method. pages 368–377.

Peter West, Ari Holtzman, Jan Buys, and Yejin Choi. 2019. BottleSum: Unsupervised and self-supervised sentence summarization using the information bottleneck principle. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3752–3761, Hong Kong, China. Association for Computational Linguistics.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8(3–4):229–256.

Zixiaofan Yang, Einolghozati Arash, Inan Hakan, Diedrick Keith, Fan Angela, Dulmez Pinar, and Gupta Sona. 2020a. Improving text-to-text pre-trained models for the graph-to-text task. In *WebNLG workshop at INLG 2020*.

Ziyi Yang, Chenguang Zhu, Robert Gmyr, Michael Zeng, Xuedong Huang, and Eric Darve. 2020b. TED: A pretrained unsupervised summarization model with theme modeling and denoising. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1865–1874, Online. Association for Computational Linguistics.

Haoyu Zhang, Jingjing Cai, Jianjun Xu, and Ji Wang. 2019a. Pretraining-based natural language generation for text summarization. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 789–797, Hong Kong, China. Association for Computational Linguistics.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2019b. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. *ArXiv*, abs/1912.08777.

Xingxing Zhang, Furu Wei, and Ming Zhou. 2019c. HIBERT: Document level pre-training of hierarchical bidirectional transformers for document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5059–5069, Florence, Italy. Association for Computational Linguistics.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019d. ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451, Florence, Italy. Association for Computational Linguistics.

Yao Zhao, M. Saleh, and Peter J. Liu. 2020. Seal: Segment-wise extractive-abstractive long-form text summarization. *ArXiv*, abs/2006.10213.