# PRIMERA: Pyramid-based Masked Sentence Pre-training for Multi-document Summarization

Anonymous ACL submission

## Abstract

We introduce PRIMERA, a pre-trained model for multi-document representation with a focus on summarization that reduces the need for dataset-specific architectures and large amounts of fine-tuning labeled data. PRIMERA uses our newly proposed pre-training objective designed to teach the model to connect and aggregate information across documents. It also uses efficient encoder-decoder transformers to simplify the processing of concatenated input documents. With extensive experiments on 6 multi-document summarization datasets from 3 different domains on zero-shot, few-shot and full-supervised settings, PRIMERA outperforms current state-of-the-art dataset-specific and pre-trained models on most of these settings with large margins.[1]

## 1 Introduction

Multi-Document Summarization is the task of generating a summary from a cluster of related documents. State-of-the-art approaches to multi-document summarization are primarily either graph-based (Liao et al., 2018; Li et al., 2020; Pasunuru et al., 2021), leveraging graph neural networks to connect information between the documents, or hierarchical (Liu and Lapata, 2019a; Fabbri et al., 2019; Jin et al., 2020), building intermediate representations of individual documents and then aggregating information across. While effective, these models either require domain-specific additional information e.g. Abstract Meaning Representation (Liao et al., 2018), or discourse graphs (Christensen et al., 2013; Li et al., 2020), or use dataset-specific, customized architectures, making it difficult to leverage pretrained language models. Simultaneously, recent pretrained language models (typically encoder-decoder transformers) have shown the advantages of pretraining and transfer learning for generation and summarization (Raffel et al., 2020; Lewis et al., 2020; Beltagy et al., 2020; Zaheer et al., 2020). Yet, existing pretrained models either use single-document pretraining objectives or use encoder-only models that do not work for generation tasks like summarization (e.g., CDLM, Caciularu et al., 2021).

[1]The code and pre-trained models will be released.

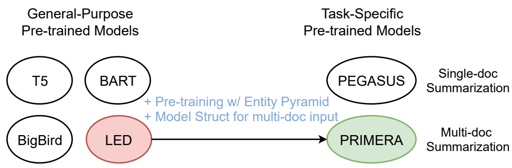


Figure 1: PRIMERA vs existing pretrained models.

Therefore, we argue that these pretrained models are not necessarily the best fit for multi-document summarization. Alternatively, we propose a simple pretraining approach for multi-document summarization, reducing the need for dataset-specific architectures and large fine-tuning labeled data (See Figure 1 to compare with other pretrained models). Our method is designed to teach the model to identify and aggregate salient information across a "cluster" of related documents during pretraining. Specifically, our approach uses the Gap Sentence Generation objective (GSG) (Zhang et al., 2020), i.e. masking out several sentences from the input document, and recovering them in order in the decoder. We propose a novel strategy for GSG sentence masking which we call, Entity Pyramid, inspired by the Pyramid Evaluation method (Nenkova and Passonneau, 2004). With Entity Pyramid, we mask salient sentences in the entire cluster then train the model to generate them, encouraging it to find important information across documents and aggregate it in one summary.

We conduct extensive experiments on **6** multi-document summarization datasets from **3** different domains. We show that despite its simplicity, PRIMERA achieves superior performance compared with prior state-of-the-art pretrained models,

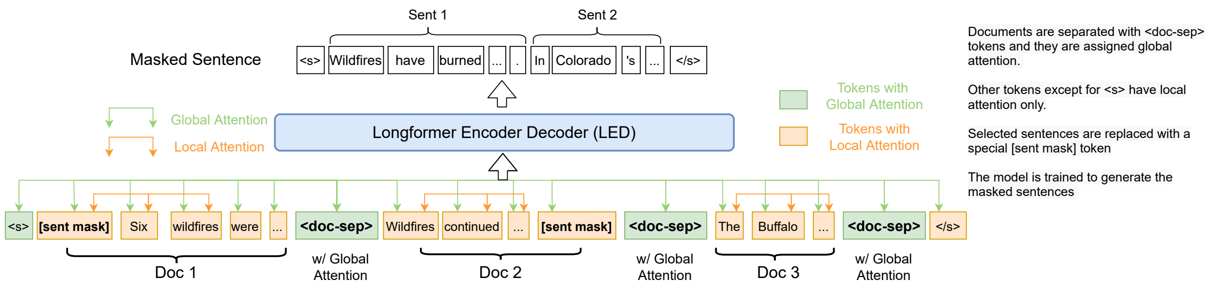


Figure 2: Model Structure of PRIMERA.

as well as dataset-specific models in both few-shot and full fine-tuning settings. PRIMERA performs particularly strong in zero- and few-shot settings, significantly outperforming prior state-of-the-art up to **5** Rouge-1 points with as few as 10 examples. Our contributions are summarized below:

1. We release PRIMERA, the first pretrained generation model for multi-document inputs with focus on summarization.
2. We propose Entity Pyramid, a novel pretraining strategy that trains the model to select and aggregate salient information from documents.
3. We extensively evaluate PRIMERA on 6 datasets from 3 different domains for zero-shot, few-shot and fully-supervised settings. We show that PRIMERA outperforms current state-of-the-art on most of these evaluations with large margins.

## 2 Model

In this section, we discuss our proposed model PRIMERA, a new pretrained general model for multi-document summarization. Unlike prior work, PRIMERA minimizes dataset-specific modeling by simply concatenating a set of documents and processing them with a general efficient encoder-decoder transformer model (§2.1). The underlying transformer model is pretrained on an unlabeled multi-document dataset, with a new entity-based sentence masking objective to capture the salient information within a set of related documents (§2.2).

### 2.1 Model Architecture and Input Structure

Our goal is to minimize dataset-specific modeling to leverage general pretrained transformer models for the multi-document task and make it easy to use in practice. Therefore, to summarize a set of related documents, we simply concatenate all the documents in a single long sequence, and process them with an encoder-decoder transformer model. Since the concatenated sequence is long, instead of more standard encoder-decoder transformers like BART (Lewis et al., 2020) and T5 (Raffel et al., 2020), we use the Longformer-Encoder-Decoder (LED) Model (Beltagy et al., 2020), an efficient transformer model with linear complexity with respect to the input length.[2] LED uses a sparse local+global attention mechanism in the encoder self-attention side while using the full attention on decoder and cross-attention.

When concatenating, we add special document separator tokens (`<doc-sep>`) between the documents to make the model aware of the document boundaries (Figure 2). We also assign global attention to these tokens which the model can use to share information across documents (Caciularu et al., 2021) (see §5 for ablations of the effectiveness of this input structure and global attention).

### 2.2 Pretraining objective

In summarization, task-inspired pretraining objectives have been shown to provide gains over general-purpose pretrained transformers (PEGASUS; Zhang et al., 2020). In particular, PEGASUS introduces Gap Sentence Generation (GSG) as a pretraining objective where some sentences are masked in the input and the model is tasked to generate them. Following PEGASUS, we use the GSG objective, but introduce a new masking strategy designed for multi-document summarization. As in GSG, we select and mask out $m$ summary-like sentences from the input documents we want to summarize, i.e. every selected sentence is replaced by a single token `[sent-mask]` in the input, and train the model to generate the concatenation of those

[2] We use LED and not other efficient transformers like Bigbird-PEGASUS (Zaheer et al., 2020) for two reasons, the first is that BigBird's global attention can't be assigned to individual tokens in the middle of the sequence, which is important for the representation of long documents as shown in Caciularu et al. (2021). Second, because pretrained checkpoints are available for LED, while BigBird-PEGASUS released the already fine-tuned checkpoints.

sentences as a "pseudo-summary" (Figure 2). This is close to abstractive summarization because the model needs to reconstruct the masked sentences using the information in the rest of the documents.

The key idea is how to select sentences that best summarize or represent a set of related input documents (which we also call a "cluster"), not just a single document as in standard GSG. Zhang et al. (2020) use three strategies - Random, Lead (first $m$ sentences), and "Principle". The "Principle" method computes sentence salience score based on ROUGE score of each sentence, $s_i$, w.r.t the rest of the document ($D/\{s_i\}$), i.e. $\text{Score}(s_i) = \text{ROUGE}(s_i, D/\{s_i\})$. Intuitively, this assigns a high score to the sentences that have a high overlap with the other sentences.

However, we argue that a naive extension of such strategy to multi-document summarization would be sub-optimal since multi-document inputs typically include redundant information, and such strategy would prefer an exact match between sentences, resulting in a selection of less representative information (see Appx. G for an example of sentences selected by the Principle strategy.).

To address this limitation, we propose a new masking strategy inspired by the Pyramid Evaluation framework (Nenkova and Passonneau, 2004) which was originally developed for evaluating summaries with multiple human written references. Our strategy aims to select sentences that best represent the entire cluster of input documents.

### 2.2.1 Entity Pyramid Masking

**Pyramid Evaluation** The Pyramid Evaluation method (Nenkova and Passonneau, 2004) is based on the intuition that relevance of a unit of information can be determined by the number of references (i.e. gold standard) summaries that include it. The unit of information is called Summary Content Unit (SCU); words or phrases that represent single facts. These SCUs are first identified by human annotators in each reference summary, and they receive a score proportional to the number of reference summaries that contain them. A Pyramid Score for a candidate summary is then the normalized mean of the scores of the SCUs that it contains. One advantage of the Pyramid method is that it directly assesses the content quality.

**Entity Pyramid Masking** Inspired by how content saliency is measured in the Pyramid Evaluation, we hypothesize that a similar idea could be applied for the multi-document summarization to identify salient sentences for masking. Specifically, for a cluster with multiple related documents, the more documents an SCU appears in, the more salient that information should be to the cluster. Therefore, it should be considered for inclusion in the pseudo-summary in our masked sentence generation objective. SCUs in the original Pyramid Evaluation are human-annotated, which is not feasible for large scale pretraining. As a proxy, we explore leveraging information expressed as named entities, since they are key building blocks in extracting information from text about events/objects and the relationships between their participants/parts (Jurafsky and Martin, 2009). Following the Pyramid framework, we use the entity frequency in the cluster as a proxy for saliency. Concretely, as shown in Fig. 3, we have the following three steps to select salient sentences in our masking strategy:

1. *Entity Extraction*. We extract named entities using SpaCy (Honnibal et al., 2020).[3]
2. *Entity Pyramid Estimation*. We then build an Entity Pyramid for estimating the salience of entities based on their document frequency, i.e. the number of documents each entity appears in.
3. *Sentence Selection*. Similar to the Pyramid evaluation framework, we identify salient sentences with respect to the cluster of related documents. Algorithm 1 shows the sentence selection procedure. As we aim to select the entities better representing the whole cluster instead of a single document, we first remove all entities from the Pyramid that appear only in one document. Next, we iteratively select entities from top of the pyramid to bottom (i.e., highest to lowest frequency), and then select sentences in the document that include the entity as the

**Algorithm 1** Entity Pyramid Sentence Selection

**Input:** Document cluster
**Input:** List of entities w/ frequency $> 1$. $N$ length of the list
**Input:** $m$ number of sentences to select
**Output:** List of sentences to mask

```
E ← sort entities by frequency, descending
selected = []
for i ← 1 to |E| do
    SentCand ← all sentences in the cluster containing E[i]
    cur_sent = arg max_{s∈SentCand} Score(s)
    selected.append(cur_sent)
    if |selected| == m then
        Break
    end if
end for
Return selected
```

[3]Note that entity information is only used at pretraining time. This is unlike some prior work that utilize additional information (like named entities, coref, discourse, or AMR) at fine-tuning and inference time.

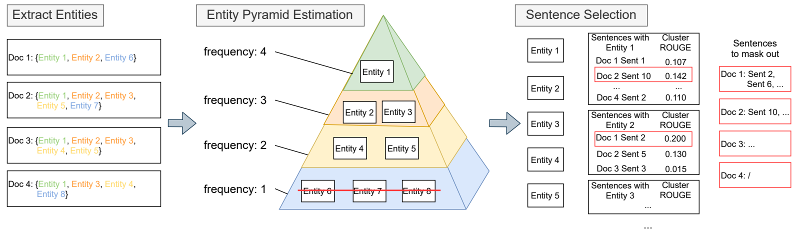


Figure 3: The Entity Pyramid Strategy to select salient sentences for masking. Pyramid entity is based on the frequency of entities in the documents. The most representative sentence are chosen based on Cluster ROUGE for each entity with frequency $> 1$, e.g. Sentence 10 in Document 2 for Entity 1.

initial candidate set. Finally, within this candidate set, we find the most representative sentences to the cluster by measuring the content overlap of the sentence w.r.t documents other than the one it appears in. This final step supports the goal of our pre-training objective, namely to reconstruct sentences that can be recovered using information from other documents in the cluster, which encourages the model to better connect and aggregate information across multiple documents. Following Zhang et al. (2020) we use ROUGE scores (Lin, 2004) as a proxy for content overlap. For each sentence $s_i$, we specifically define a Cluster ROUGE score as $Score(s_i) = \sum_{\{doc_j \in C, s_i \notin doc_j\}} \mathrm{ROUGE}(s_i, doc_j)$ Where $C$ is the cluster of related documents.
Note that different from the importance heuristic defined in PEGASUS (Zhang et al., 2020), Entity Pyramid strategy favors sentences that are representative of more documents in the cluster than the exact matching between fewer documents(Appx. §G shows a qualitative example.).The benefit of our strategy is shown in an ablation study (§5).

## 3 Experiment Goals

We aim to answer the following questions:
- Q1: How does PRIMERA perform, compared with existing pre-trained generation models in zero- and few-shot settings? See §4.2.
- Q2: How does PRIMERA perform, compared with current state-of-the-art models, in the fully supervised setting? See §4.5.
- Q3: How much is the contribution of each component in PRIMERA, i.e. input structure, pretraining, and masking strategy? See §5.
- Q4: What is the effect of our entity pyramid strategy, compared with the strategy used in PEGASUS? See §5
- Q5: Is PRIMERA able to capture salient information and generate fluent summaries? See §6.

With these goals, we explore the effectiveness of PRIMERA quantitatively on multi-document summarization benchmarks, verify the improvements by comparing PRIMERA with multiple existing pre-trained models and SOTA models, and further validate the contribution of each component with carefully controlled ablations. An additional human evaluation is conducted to show PRIMERA is able to capture salient information and generate more fluent summaries.

| Dataset | #Examples | #Doc/C | $Len_{src}$ | $Len_{summ}$ |
|---|---|---|---|---|
| Newshead (2020) | 360K | 3.5 | 1734 | - |
| Multi-News (2019) | 56K | 2.8 | 1793 | 217 |
| Multi-Xscience (2020) | 40K | 4.4 | 700 | 105 |
| Wikisum* (2018) | 1.5M | 40 | 2238 | 113 |
| WCEP-10 (2020) | 10K | 9.1 | 3866 | 28 |
| DUC2004 (2005) | 50 | 10 | 5882 | 115 |
| arXiv (2018) | 214K | 5.5 | 6021 | 272 |

Table 1: The statistics of all the datasets we explore in this paper. *We use subsets of Wikisum (10/100, 3200) for few-shot training and testing only.

## 4 Experiments

### 4.1 Experimental Setup

**Implementation Details** We use the Longformer-Encoder-Decoder (LED) (Beltagy et al., 2020) large as our model initialization, The length limits of input and output are 4096 and 1024, respectively, with sliding window size as $w = 512$ for local attention in the input. (More implementation details of pretraining process can be found in Appx §A)
**Pretraining corpus** For pretraining, we use the *Newshead* dataset (Gu et al., 2020) (row 1, Table 1), a relatively large resource, where each instance includes a set of related documents about a specific news event. Note that this dataset does not have any ground-truth summaries.
**Evaluation Datasets** We evaluate our approach on wide variety of multi-document summarization datasets plus one single document dataset from

| Models | Multi-News(256) | | | Multi-XSci(128) | | | WCEP(50) | | | WikiSum(128) | | | arXiv(300) | | | DUC2004 (128) | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R-1 | R-2 | R-L | R-1 | R-2 | R-L | R-1 | R-2 | R-L | R-1 | R-2 | R-L | R-1 | R-2 | R-L | R-1 | R-2 | R-L |
| PEGASUS⋆(Zhang et al., 2020) | 36.5 | 10.5 | 18.7 | - | - | - | - | - | - | - | - | - | 28.1 | 6.6 | 17.7 | - | - | - |
| PEGASUS (our run) | 32.0 | 10.1 | 16.7 | 27.6 | **4.6** | 15.3 | **33.2** | **12.7** | **23.8** | 24.6 | 5.5 | 15.0 | 29.5 | 7.9 | 17.1 | 32.7 | **7.4** | 17.6 |
| BART (our run) | 27.3 | 6.2 | 15.1 | 18.9 | 2.6 | 12.3 | 20.2 | 5.7 | 15.3 | 21.6 | 5.5 | 15.0 | 29.2 | 7.5 | 16.9 | 24.1 | 4.0 | 15.3 |
| LED (our run) | 17.3 | 3.7 | 10.4 | 14.6 | 1.9 | 9.9 | 18.8 | 5.4 | 14.7 | 10.5 | 2.4 | 8.6 | 15.0 | 3.1 | 10.8 | 16.6 | 3.0 | 12.0 |
| PRIMERA (our model) | **42.0** | **13.6** | **20.8** | **29.1** | 4.6 | **15.7** | 28.0 | 10.3 | 20.9 | **28.0** | **8.0** | **18.0** | **34.6** | **9.4** | **18.3** | **35.1** | 7.2 | **17.9** |

Table 2: Zero-shot results. The models in the first block use the full-length attention ($O(n^2)$) and are pretrained on the single document datasets. The numbers in the parenthesis following each dataset indicate the output length limit set for inference. PEGASUS⋆ means results taken exactly from PEGASUS (Zhang et al., 2020), where available.

various domains (News, Wikipedia, and Scientific literature). See Table 1 for dataset statistics and Appx. §B for details of each dataset.

**Evaluation metrics** Following previous works (Zhang et al., 2020), we use ROUGE scores (R-1, -2, and -L), which are the standard evaluation metrics, to evaluate the downstream task of multi-document summarization.[4] For better readability, we use AVG ROUGE scores (R-1, -2, and -L) for evaluation in the few-shot setting.

### 4.2 Zero- and Few-shot Evaluation

Many existing works in adapting pretrained models for summarization require large amounts of fine-tuning data, which is often impractical for new domains. In contrast, since our pretraining strategy is mainly designed for multi-document summarization, we expect that our approach can quickly adapt to new datasets without the need for significant fine-tuning data. To test this hypothesis, we first provide evaluation results in zero and few-shot settings where the model is provided with no, or only a few (10 and 100) training examples. Obtaining such a small number of examples should be viable in practice for new datasets.

**Comparison** To better show the utility of our pretrained models, we compare with three state-of-the-art pretrained generation models , i.e. BART (Lewis et al., 2020)[5], PEGASUS (Zhang et al., 2020) and Longformer-Encoder-Decoder(LED) (Beltagy et al., 2020). These pretrained models have been shown to outperform dataset-specific models in summarization (Lewis et al., 2020; Zhang et al., 2020), and because of pretraining, they are expected to also work well in the few-shot settings. As there is no prior work doing few-shot and zero-shot evaluations on all the datasets we consider, and also the results in the few-shot setting might be influenced by sampling variability (especially with only 10 examples) (Bragg et al., 2021), we run the same experiments for the compared models five times with different random seeds (shared with all the models), with the publicly available checkpoints .[6]

Similar to Pasunuru et al. (2021), the inputs of all the models are the concatenations of the documents within the clusters (in the same order), each document is truncated based on the input length limit divided by the total number of documents so that all documents are represented in the input. [7] To preserve the same format as the corresponding pretrained models

We set the length limit of output for BART and PEGASUS exactly as their pretrained settings on all of the datasets (Except for the zero-shot experiments, the details can be found in Sec.4.3). Regarding length limit of inputs, we tune the baselines by experimenting with 512, 1024, 4096 on Multi-News dataset in few-shot setting (10 data examples), and the model with length limit 512(PEGASUS)/1024(BART) achieves the best performance, thus we use this setting. (Detailed experiment results for different input lengths can be found in Appx. §C.1) We use the same length limit as our model for the LED model, i.e. 4096/1024 for input and output respectively, for all the datasets.

### 4.3 Zero-Shot Results

For zero-shot[8] abstractive summarization experiments, since the models have not been trained on the downstream datasets, the lengths of generated summaries mostly depend on the pretrained set-

[4] We use https://github.com/google-research/google-research/tree/master/rouge with default stemmer settings.

[5] Pilot experiments comparing BART and T5 showed BART to outperform T5 on the few-shot evaluation of Multi-News (with AVG ROUGE of 23.5/26.4 (T5) v.s. 25.2/26.7 (BART) for 10/100 training examples, respectively). Thus, we are using BART as one of the baselines.

[6] Checkpoints from https://huggingface.co/models

[7] pilot experiments show simple truncation results in inferior performance, which is in line with Pasunuru et al. (2021).

[8] For clarity, by zero-shot we mean using the pretrained model directly without any additional supervision

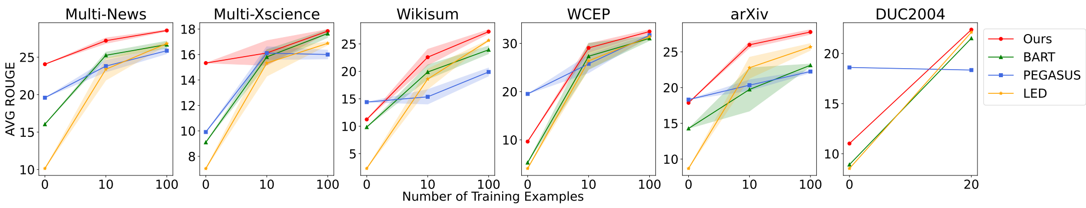


Figure 4: The AVG ROUGE scores (R-1, R-2 and R-L) of the pretrained models with 0, 10 and 100 training data with variance. All the results of few-shot experiments (10 and 100) are obtained by the average of 5 random runs (with std, and the same set of seeds shared by all the models).

tings. Thus to better control the length of generated summaries and for a fair comparison between all models, following Zhu et al. (2019), we set the length limit of the output at inference time to the average length of gold summaries.[9] Exploring other approaches to controlling length at inference time (e.g., Wu et al., 2021) is an orthogonal direction which we leave for future work.

Table 2 shows the performance comparison among all the models. Results indicate that our model achieves substantial improvements compared with all the three baselines on most of the datasets. As our model is pretrained on clusters of documents with longer input and output, the benefit is stronger on the dataset with longer summaries, e.g. Multi-News and arXiv. Comparing PEGASUS and BART models, as the objective of PEGASUS is designed mainly for summarization tasks, not surprisingly it has relatively better performances across different datasets. Interestingly, LED underperforms other models, plausibly since part of the position embeddings (1k to 4k) are not pretrained. Encouragingly, our model performs the best, demonstrating the benefits of our pretraining strategy for multi-document summarization

### 4.4 Few Shot Evaluation

Compared with the strict zero-shot scenario, few-shot experiments are closer to the practical scenarios, as it is arguably affordable to label dozens of examples for almost any application.

We fine-tune all of the four models on different subsets with 10 and 100 examples, and the results are shown in Figure 4. (hyperparameter settings in Appx. §D.1) Since R-1, -2, and -L show the same trend, we simply show the average of the three metrics in the figure for brevity (full ROUGE scores can be found in Appx. Table 8) To show the generality, all the results of few-shot experiments are the average over 5 runs on different subsets (shared by all the models).

The result of each run is obtained by the 'best' model chosen based on the ROUGE scores on a randomly sampled few-shot validation set with the same number of examples as the training set, which is similar with Zhang et al. (2020). Note that their reported best models have been selected based on the whole validation set which may give PEGASUS some advantage. Nevertheless, we argue that sampling few-shot validation sets as we do here is closer to real few-shot scenarios (Bragg et al., 2021).

Our model outperforms all baselines on all of the datasets with 10 and 100 examples demonstrating the benefits of our pretraining strategy and input structure. Comparing the performances of our model with the different number of training data fed in, our model converges faster than other models with as few as 10 data examples.

### 4.5 Fully Supervised Evaluation

To show the advantage of our pretrained model when there is abundant training data, we also train the model with the full training set (hyperparameter settings can be found in Appx. §D.2). Table 3 shows the performance comparison with previous state-of-the-art[10], along with the results of previous SOTA. We observe that PRIMERA achieves state-of-the-art results on Multi-News, WCEP, and arXiv, while slightly underperforming the prior work on Multi-XScience (R-1). On Multi-XScience clusters have less overlapping information which is slightly different than the pretraining setting of PRIMERA. The source documents in this dataset are the abstracts of all the publications cited in the related work paragraphs, which might be less similar to

[9] In practice, it is reasonable to assume knowing the approximate length of the expected summary for a given task/domain.

[9] We re-evaluate the generated summaries of the models from Lu et al. (2020) for Multi-XScience, as we use a different version of ROUGE.

[10] Due to the lack of computational resources, we do not train the model on Wikisum.

| Datasets | Previous SOTA | | | PRIMERA | | |
|---|---|---|---|---|---|---|
| | R-1 | R-2 | R-L | R-1 | R-2 | R-L |
| Multi-News | 49.2 | 19.6 | 24.5 | **49.9** | **21.1** | **25.9** |
| Multi-XScience | **33.9** | 6.8 | **18.2** | 31.9 | **7.4** | 18.0 |
| WCEP | 35.4 | 15.1 | 25.6 | **46.1** | **25.2** | **37.9** |
| arXiv | 46.6 | 19.6 | 41.8 | **47.6** | **20.8** | **42.6** |

Table 3: Fully supervised results. Previous SOTA are from Pasunuru et al. (2021) for Multi-News, Lu et al. (2020) for Multi-XScience[11], Hokamp et al. (2020) for WCEP, and Beltagy et al. (2020) for arXiv.

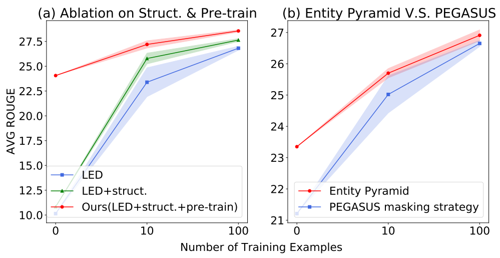


Figure 5: Ablation study with the few-shot setting on the Multi-News dataset regarding to (a) input Structure (`<doc-sep>` tokens between documents and global attention on them) and pretraining, (b) pretraining using PEGASUS vs our approach.

each other and the target related work. PRIMERA outperforms the LED model (State-of-the-art) on the arXiv dataset while using a sequence length 4x shorter (4K in PRIMERA v.s. 16K in LED), further showing that the pretraining and input structure of our model not only works for multi-document summarization but can be also effective for summarizing single documents having multiple sections.

## 5 Ablation Study

We conduct ablation studies on the Multi-News dataset in few-shot setting, to validate the contribution of each component in our pretrained models.

**Input structure** In Figure 5 (a) we observe the effectiveness of both pretraining and the input structure (`<doc-sep>` tokens between documents and global attention on them)

**Sentence masking strategy** To isolate the effect of our proposed pretraining approach, we compare with an exact model architecture when pretrained on the same amount of data using the PEGASUS (Zhang et al., 2020) masking strategy instead of ours. We keep all the other settings the same (e.g., data, length limit of input and output, pretraining dataset, input structure, as well as the separators) and only modify the pretraining masking strategy. We run the same experiments under zero-/few-shot scenarios on the Multi-News dataset as in §4.2, and

| Model | DUC2007(20) | | | | TAC2008(20) | | | |
|---|---|---|---|---|---|---|---|---|
| | $S_r$ | R | P | F | $S_r$ | R | P | F |
| PEGASUS | 6.0 | 2.5 | 2.4 | 2.4 | **8.7** | **9.1** | 9.4 | 9.1 |
| LED | 9.6 | 3.9 | 4.0 | 3.8 | 6.9 | 7.1 | **10.8** | 8.4 |
| PRIMERA | **12.5** | **5.1** | **5.0** | **5.0** | 8.5 | 8.9 | 10.0 | **9.3** |

Table 4: Pyramid Evaluation results: Raw scores $S_r$, (R)ecall, (P)recision and (F)-1 score. For readability, Recall, Precision and F-1 scores are multiplied by 100.

the results are shown in Figure 5 (b). The model pretrained with our Entity Pyramid strategy shows a clear improvement under few-shot scenarios.

## 6 Human Evaluation

We also conduct human evaluations to validate the effectiveness of PRIMERA on DUC2007 and TAC2008 (Dang and Owczarzak, 2008) datasets in the few-shot setting (10/10/20 examples for train/valid/test). Both datasets consist of clusters of news articles, and DUC2007 contains longer inputs (25 v.s. 10 documents/cluster) and summaries (250 v.s. 100 words). Since the goal of our method is to enable the model to better aggregate information across documents, we evaluate the content quality of the generated summaries following the original Pyramid human evaluation framework (Nenkova and Passonneau, 2004). In addition, we also evaluate the fluency of generated summaries following the DUC guidelines.[12] Details can be found in Appx. H.

**Compared Models** We compare our model with LED and PEGASUS in human evaluations. Because PEGASUS is a task-specific model for abstractive summarization, and LED has the same architecture and length limits as our model with the parameters inherited from BART, which is more comparable with our model than vanilla BART.

**Pyramid Evaluation** Both TAC and DUC datasets include SCU (Summary Content Unit) annotations and weights identified by experienced annotators. We then ask 3 annotators to make a binary decision whether each SCU is covered in a candidate summary. Following Nenkova and Passonneau (2004), the raw score of each summary is then computed by the sum of weights of the covered SCUs, i.e. $S_r = \sum_{SCU} w_i I(SCU_i)$, where $I(SCU_i)$ is an indicator function on whether $SCU_i$ is covered by the current summary, and $w_i$ is the

[12] https://www-nlpir.nist.gov/projects/duc/duc2007/quality-questions.txt

| Model | DUC2007(20) | | | TAC2008(20) | | |
|---|---|---|---|---|---|---|
| | Gram. | Ref. | Str.&Coh. | Gram. | Ref. | Str.&Coh. |
| PEGASUS | 4.45 | 4.35 | 1.95 | **4.40** | 4.20 | 3.20 |
| LED | 4.35 | 4.50 | 3.20 | 3.10 | 3.80 | 2.55 |
| PRIMERA | **4.70** | **4.65** | **3.70** | **4.40** | **4.45** | **4.10** |

Table 5: The results of Fluency Evaluation on two datasets, in terms of the Grammaticality , Referential clarity and Structure & Coherence.

weight of $SCU_i$. And considering the length of generated summaries, we further compute Recall, Precision and F-1 score regarding lengths of both gold and system generated summaries.[13] As shown in Table 4, PRIMERA achieves the best F-1 score.

**Fluency Evaluation** Fluency results can be found in Table 5, and PRIMERA has the best performance on both datasets in terms of all aspects.

## 7 Related Work

**Neural Multi-Document Summarization** These models can be categorized into two classes, graph-based models (Yasunaga et al., 2017; Liao et al., 2018; Li et al., 2020; Pasunuru et al., 2021) and hierarchical models (Liu and Lapata, 2019a; Fabbri et al., 2019; Jin et al., 2020). Graph-based models often require auxiliary information (e.g., AMR, discourse structure) to build an input graph, making them reliant on auxiliary models and less general. Hierarchical models are another class of models for multi-document summarization, examples of which include multi-head pooling and inter-paragraph attention (Liu and Lapata, 2019a), MMR-based attention (Fabbri et al., 2019; Mao et al., 2020), and attention across representations of different granularity (words, sentences, and documents) (Jin et al., 2020). Such models are often dataset-specific and difficult to develop and adapt to other datasets or tasks.

**Pretrained Models for Summarization** Pretrained language models have been successfully applied to summarization, e.g., BERTSUM (Liu and Lapata, 2019b), BART (Lewis et al., 2020), T5 (Raffel et al., 2020). Instead of regular language modeling objectives, PEGASUS (Zhang et al., 2020) introduced a pretraining objective with a focus on summarization, using Gap Sentence Generation, where the model is tasked to generate summary-worthy sentences, and Zou et al. (2020) proposed different pretraining objectives to reinstate the original document, specifically for summarization task as well. Contemporaneous work by Rothe et al. (2021) argued that task-specific pretraining does not always help for summarization, however, their experiments are limited to single-document summarization datasets. Pretraining on the titles of HTMLs has been recently shown to be useful for few-shot short-length single-document summarization as well (Aghajanyan et al., 2021). Goodwin et al. (2020) evaluate three state-of-the-art models (BART, PEGASUS, T5) on several multi-document summarization datasets with low-resource settings, showing that abstractive multi-document summarization remains challenging. Efficient pretrained transformers (e.g., Longformer (Beltagy et al., 2020) and BigBird (Zaheer et al., 2020) that can process long sequences have been also proven successful in summarization, typically by the ability to process long inputs, connecting information across the entire sequence. CDLM (Caciularu et al., 2021) is a follow-up work for pretraining the Longformer model in a cross-document setting using global attention on masked tokens during pretraining. However, this model only addresses encoder-only tasks and it is not suitable for generation. In this work, we show how efficient transformers can be pretrained using a task-inspired pretraining objective for multi-document summarization.

[13]The details can be found in Appx. H

## 8 Conclusion and Future Work

We present PRIMERA a pre-trained model for multi-document summarization. Unlike prior work, PRIMERA minimizes dataset-specific modeling by using a Longformer model pretrained with a novel entity-based sentence masking objective. The pretraining objective is designed to help the model connect and aggregate information across input documents. PRIMERA outperforms prior state-of-the-art pre-trained and dataset-specific models on 6 datasets from 3 different domains, on zero, few-shot, and full fine-tuning setting. PRIMERA's top performance is also revealed by human evaluation.

In zero-shot setting, we can control the output length of generated summaries at inference time by specifying a length limit during decoding. Exploring a controllable generator in which the desired length can be injected as part of the input is a natural future direction. Besides the summarization task, we would like to explore using PRIMERA for other generation tasks with multiple documents as input, like multi-hop question answering.

## Ethics Concern

While there is limited risk associated with our work, similar to existing state-of-the-art generation models, there is no guarantee that our model will always generate factual content. Therefore, caution must be exercised when the model is deployed in practical settings. Factuality is an open problem in existing generation models.

## References


Armen Aghajanyan, Dmytro Okhonko, Mike Lewis, Mandar Joshi, Hu Xu, Gargi Ghosh, and Luke Zettlemoyer. 2021. HTLM: hyper-text pre-training and prompting of language models. *CoRR*, abs/2107.06955.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *CoRR*, abs/2004.05150.

Jonathan Bragg, Arman Cohan, Kyle Lo, and Iz Beltagy. 2021. Flex: Unifying evaluation for few-shot nlp. *arXiv preprint arXiv:2107.07170*.

Avi Caciularu, Arman Cohan, Iz Beltagy, Matthew E. Peters, Arie Cattan, and Ido Dagan. 2021. Cross-document language modeling. *ArXiv*, abs/2101.00406.

Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2013. Towards coherent multi-document summarization. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1163–1173, Atlanta, Georgia. Association for Computational Linguistics.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621, New Orleans, Louisiana. Association for Computational Linguistics.

Hoa Trang Dang. 2005. Overview of duc 2005. In *Proceedings of the document understanding conference*, volume 2005, pages 1–12.

Hoa Trang Dang and Karolina Owczarzak. 2008. Overview of the tac 2008 update summarization task. *Theory and Applications of Categories*.

Alexander Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1074–1084, Florence, Italy. 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.

Demian Gholipour Ghalandari, Chris Hokamp, Nghia The Pham, John Glover, and Georgiana Ifrim. 2020. A large-scale multi-document summarization dataset from the Wikipedia current events portal. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1302–1308, Online. Association for Computational Linguistics.

Travis Goodwin, Max Savery, and Dina Demner-Fushman. 2020. Flight of the PEGASUS? comparing transformers on few-shot and zero-shot multi-document abstractive summarization. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5640–5646, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Xiaotao Gu, Yuning Mao, Jiawei Han, Jialu Liu, You Wu, Cong Yu, Daniel Finnie, Hongkun Yu, Jiaqi Zhai, and Nicholas Zukoski. 2020. Generating representative headlines for news stories. In *Proceedings of The Web Conference 2020*, WWW '20, page 1773–1784, New York, NY, USA. Association for Computing Machinery.

Chris Hokamp, Demian Gholipour Ghalandari, Nghia The Pham, and John Glover. 2020. Dyne: Dynamic ensemble decoding for multi-document summarization. *CoRR*, abs/2006.08748.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Hanqi Jin, Tianming Wang, and Xiaojun Wan. 2020. Multi-granularity interaction network for extractive and abstractive multi-document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6244–6254, Online. Association for Computational Linguistics.

Dan Jurafsky and James H. Martin. 2009. *Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition*. Pearson Prentice Hall, Upper Saddle River, N.J.

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.

Wei Li, Xinyan Xiao, Jiachen Liu, Hua Wu, Haifeng Wang, and Junping Du. 2020. Leveraging graph to improve abstractive multi-document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6232–6243, Online. Association for Computational Linguistics.

Kexin Liao, Logan Lebanoff, and Fei Liu. 2018. Abstract Meaning Representation for multi-document summarization. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1178–1190, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

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

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. 2019a. Hierarchical transformers for multi-document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5070–5081, Florence, Italy. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2019b. 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.

Yao Lu, Yue Dong, and Laurent Charlin. 2020. Multi-XScience: A large-scale dataset for extreme multi-document summarization of scientific articles. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8068–8074, Online. Association for Computational Linguistics.

Yuning Mao, Yanru Qu, Yiqing Xie, Xiang Ren, and Jiawei Han. 2020. Multi-document summarization with maximal marginal relevance-guided reinforcement learning. *arXiv preprint arXiv:2010.00117*.

Ani Nenkova and Rebecca Passonneau. 2004. Evaluating content selection in summarization: The pyramid method. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 145–152, Boston, Massachusetts, USA. Association for Computational Linguistics.

Ramakanth Pasunuru, Mengwen Liu, Mohit Bansal, Sujith Ravi, and Markus Dreyer. 2021. Efficiently summarizing text and graph encodings of multi-document clusters. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4768–4779, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Sascha Rothe, Joshua Maynez, and Shashi Narayan. 2021. A thorough evaluation of task-specific pretraining for summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 140–145, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Chien-Sheng Wu, Linqing Liu, Wenhao Liu, Pontus Stenetorp, and Caiming Xiong. 2021. Controllable abstractive dialogue summarization with sketch supervision. *arXiv preprint arXiv:2105.14064*.

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

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences. *CoRR*, abs/2007.14062.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 11328–11339. PMLR.

Chenguang Zhu, Ziyi Yang, Robert Gmyr, Michael Zeng, and Xuedong Huang. 2019. Make lead bias in your favor: Zero-shot abstractive news summarization. *arXiv: Computation and Language*.

Yanyan Zou, Xingxing Zhang, Wei Lu, Furu Wei, and Ming Zhou. 2020. Pre-training for abstractive document summarization by reinstating source text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3646–3660, Online. Association for Computational Linguistics.

## A Implementation details of pre-training

As the multi-document summarization task has a higher compression ratio, defined as $len(Summary)/len(Input)$, (e.g. 12% for Multi-News dataset and 15% for Multi-Xscience dataset), we use 15% as the ratio of masked sentences for generation. In addition to this 15% masked sentences, following PEGASUS (Zhang et al., 2020), we also copy an additional 15% of the input sentences to the output without masking them in the input. This allows the model to also learn to copy information from the source directly and found to be useful by Zhang et al. (2020).

We pretrain the model for 100K steps, with early stopping, batch size of 16, Adam optimizer with a learning rate of $3e{-}5$ following Beltagy et al. (2020), with 10K warmup steps and linear decay. The pretraining process takes likely 7 days on 4 A100 GPUs.

As the backbone of PRIMERA is the Longformer Encoder Decoder model (LED), it has the same number of parameters with LED (447M).

## B Detailed Description on the Evaluation Datasets

The details of evaluation datasets can be found below.

*Multi-News* (Fabbri et al., 2019): A multi-document dataset with summaries written by professional editors from the newser.com.

*Wikisum* (Liu* et al., 2018) Each summary is a Wikipedia article, and the source documents are either citations in the reference section or the Web Search results of section titles.[14] In our experiments, we use the data crawled by Liu and Lapata (2019a).

*WCEP* (Gholipour Ghalandari et al., 2020) is built based on news events from Wikipedia Current Events Portal and the references are obtained similar to Wikisum. There are at most 100 documents within each cluster in the original dataset, thus we remove all the duplicates and only keep up to 10 documents for each cluster based on the relevance score in the original dataset, which is similar to the WCEP-10 variant in the original paper.

*Multi-X-Science* (Lu et al., 2020) a multi-document summarization dataset created from scientific articles, the summaries are paragraphs of related work section, while source documents include the abstracts of the query and referred papers.

*DUC* benchmarks (Dang, 2005) include multi-document summarization datasets in the news domain, with 10-30 documents and 3-4 human-written summaries per cluster. Since these datasets are small, we use them primarily for a few-shot evaluation. We use DUC2003 for training (only one of the reference summaries for each document is used for training) and DUC2004 as test.

*ArXiv* (Cohan et al., 2018) is a single document summarization dataset in the scientific paper domain. Each document is a scientific paper, and the summary is the corresponding abstract. As each scientific paper consists of multiple sections, we treat each section as a separate document within a cluster in our experiments. This is to evaluate our model's effectiveness on summarizing single documents having multiple sections.

## C Details on Compared models

The details of compared models in the zero-/few-shot setting can be found below.

*BART* (Lewis et al., 2020) an encoder-decoder transformer model pretrained on the objective of reconstructing the corrupted documents in multiple ways, e.g. Token Deletion, Text Infilling, Sentence Rotation and etc.

PEGASUS (Zhang et al., 2020) a pretrained model designed for abstractive summarization as the downstream task, especially for the single document input. It is trained on the objective of Gap Sentence Generation on C4 (Raffel et al., 2020) and Hugenews datasets (Note that the pretraining data size in PEGASUS is magnitudes larger than ours). As it is only evaluated on one multi-document summarization dataset (Multi-news), we rerun the model on all the datasets. To verify the quality of our reproduction, the average ROUGE scores of our re-run model vs. (the ones reported on the paper) with 10 examples and 100 examples fed are $23.81 \pm 0.79$ vs. (24.13) and $25.86 \pm 0.41$ vs. (25.48), with minor differences plausibly resulting from different samplings.

*Longformer Encoder-Decoder (LED)* (Beltagy et al., 2020) is the initial state of our model before pretraining. The parameters of LED are inherited from the BART model, and to enable the model to deal with longer input, the position embeddings are repeatedly copied from BART's 1K position

[14] Due to the large size of the dataset, we evaluate all the models on the first 3200 data in the test set. And in the few-shot experiments, we randomly choose few examples (10 or 100) from the training set and validation set.

| Length Limit | BART | | | PEGASUS | | |
|---|---|---|---|---|---|---|
| | R-1 | R-2 | R-L | R-1 | R-2 | R-L |
| 512 | - | - | - | **39.0** | **12.1** | **20.3** |
| 1024 | **42.3** | **13.7** | **19.7** | 37.6 | 10.7 | 18.8 |
| 4096 | 37.9 | 11.0 | 17.5 | 34.9 | 8.7 | 17.6 |

Table 6: The ROUGE score (R-1/R-2/R-3) for pretrained models (BART and PEGASUS) with different input length limit in few-shot setting (10 data example) on the multi-news dataset. The results are the average over 5 runs on different subsets (the same seeds shared with all the other models in this paper).

embeddings. It is different from our model with respect to both pretraining and input structure (document separators and global attentions), with global attention on the (<s>) token only and no document separators.

### C.1 Detailed Experiment for Input Length Limit

We run an experiment to select the proper length limit for compared pretrained models, i.e. BART and PEGASUS. Specifically, we train both models with different input length limits (512/1024/4096) in the few-shot setting (with 10 data examples) on the multi-news dataset. Similar as the few-shot experiments described in §4.2, we train each model with each specific input length limit for 5 times on different subsets, which are shared by all the models. As shown in Table 6, BART with length limit 1024 performs the best and PEGASUS with length limit 512 performs the best, thus in all our experiments, we use 1024 as the input length limit for BART and 512 for PEGASUS.

## D Hyperparameters in Few-shot and Full Supervised Experiments

### D.1 Few-shot Experiments

We use Adam as the optimizer with linear scheduled learning rate $3e-5$ for BART, LED and our model, and use the default optimization settings of the few-shot experiments from Zhang et al. (2020), i.e. AdaFactor optimizer with scheduled learning rate $5e-4$. For all the experiments with 10 examples, the batch size is 10, the models are trained for 200 steps, with warm-up as 20 steps. For the experiments with 100 examples, we use the same batch size, with the total step and warm-up step set to be 1000 and 100, respectively.

### D.2 Fully Supervised Experiments

We use Adam as the optimizer with linear scheduled learning rate $3e-5$, and batch size as 16 for all the datasets in the full supervised experiments. The number of steps and warm-up steps are set based on the size of the datasets. The details can be found in Table 7

| Dataset | Total Steps | Warmup Steps |
|---|---|---|
| Multi-News | 25k | 2.5k |
| Multi-XScience | 20k | 2k |
| WCEP | 5k | .5k |
| arXiv | 40k | 4k |

Table 7: Details of total steps and warm-up steps used in the Full Supervised experiments.

## E Detailed Results in Few-shot Setting

The exact ROUGE scores in Figure 4 are shown in Table 8.

| Model | 0 Examples | 10 Examples | 100 Examples |
|---|---|---|---|
| | Multi-News | | |
| PEGASUS | 31.97/10.06/16.74 | 39.02/12.10/20.32 | 42.99/13.50/21.10 |
| BART | 26.10/8.98/13.06 | 42.30/13.74/19.71 | 44.23/14.77/21.02 |
| LED | 16.60/4.78/9.05 | 38.86/12.48/18.82 | 44.45/14.85/21.16 |
| Ours | **39.09/13.91/19.19** | **44.02/15.54/22.03** | **46.01/16.76/22.91** |
| | Multi-Science | | |
| PEGASUS | **27.33**/4.77/**15.04** | 28.14/4.68/**15.49** | 28.01/4.09/15.89 |
| BART | 15.21/3.49/8.61 | 27.80/4.74/14.90 | 31.17/5.32/16.45 |
| LED | 11.79/2.47/6.86 | 26.57/4.05/15.36 | 29.46/4.85/16.32 |
| Ours | 26.90/**4.98**/14.09 | **28.36/4.73**/15.29 | **31.25/5.43/16.84** |
| | Wikisum | | |
| PEGASUS | **23.67/5.37/14.17** | 23.44/6.44/16.21 | 28.50/9.83/21.33 |
| BART | 15.80/4.60/9.13 | 28.95/9.88/20.80 | 32.97/13.81/25.01 |
| LED | 8.70/2.34/5.78 | 26.53/9.30/19.95 | 34.15/16.03/26.75 |
| Ours | 17.79/5.02/10.90 | **31.10/13.26/23.39** | **36.05/17.85/27.81** |
| | WCEP | | |
| PEGASUS | **27.69/10.85/20.03** | 35.60/14.84/26.84 | 42.09/19.93/33.04 |
| BART | 7.11/3.41/5.32 | 37.46/15.82/28.70 | 41.34/19.19/32.58 |
| LED | 5.69/2.19/4.32 | 36.29/15.04/27.80 | 41.83/19.46/32.92 |
| Ours | 13.50/5.30/10.11 | **38.97/17.55/30.64** | **42.96/20.53/33.87** |
| | arXiv | | |
| PEGASUS | **29.76**/7.94/**17.27** | 33.10/8.52/19.40 | 36.38/9.55/20.83 |
| BART | 23.26/7.57/12.01 | 32.53/8.70/17.98 | 37.62/10.78/20.99 |
| LED | 13.94/3.76/8.35 | 36.51/11.16/20.68 | 41.00/13.74/22.34 |
| Ours | 29.14/**8.64**/15.82 | **41.13/13.81/23.02** | **43.42/15.85/24.07** |

Table 8: Detailed ROUGE scores (R-1/R-2/R-L) on all the datasets in the few-shot setting (corresponds to Figure 4)

## F Detailed Analysis on Fully Supervised Experiments

To show the advantage of our pre-trained model when there is sufficient data, we also train the model with the full training set, and the results can be found in Table 9-12[15], along with the results from previous works. Differently from the zero-/few-shot experiments, here we report the state-of-the-art results on different datasets, as they were presented in the corresponding original papers. Since we use the same train/valid/test set as in those prior works, we can perform a fair comparison , without re-running all those extremely time-consuming experiments .

Overall, our model achieves state-of-the-art on Multi-News (see Table 9 , WCEP dataset (see Table 11) and arXiv dataset (see Table 12).

| Models | ROUGE-1 | ROUGE-2 | ROUGE-L |
|---|---|---|---|
| PEGASUS (Zhang et al., 2020) | 47.52 | 18.72 | 24.91 |
| BART-Long-Graph (Pasunuru et al., 2021) | 49.03 | 19.04 | 24.04 |
| BART-Long-Graph(1000) (Pasunuru et al., 2021) | 49.24 | 18.99 | 23.97 |
| BART-Long(1000) (Pasunuru et al., 2021) | 49.15 | 19.50 | 24.47 |
| Ours | **49.94** | **21.05** | **25.85** |

Table 9: ROUGE scores of the previous models and our fully supervised model on the Multi-News dataset. The results of PEGASUS is from Zhang et al. (2020), and the other results are from Pasunuru et al. (2021)

**Multi-News** The experiment results on Multi-News dataset can be found in Table 9. Specifically, the PEGASUS model (Zhang et al., 2020) is pre-trained on a large-scale single-document dataset with the Gap Sentence Generation objective, which is the same as ours, but with a different masking strategy, BART-Long (Pasunuru et al., 2021) uses the same model structure as ours , and BART-Long-Graph (Pasunuru et al., 2021) additionally has discourse graph injected. Comparing the results with the BART-Long model, our model is around 1 ROUGE point higher, which may result from either better model structure or pre-training. Interestingly, in one of the ablation studies in Pasunuru et al. (2021), they find that the BART-Long model achieves its best performance with the length limit of 1000, and no further improvement is found when the length limit is greater than that. Thus we may conclude the gap between the performances is mainly from our design on the model, i.e. the document separators, proper global attention as well as the pre-training on a multi-document dataset.

**WCEP** As for the WCEP dataset, BERTREG (Gholipour Ghalandari et al., 2020) is a Regression-based sentence ranking system with BERT embedding, which is used as extractive summarization method, while Submodular+Abs is a simple two-step abstractive summarization model with a submodular-based extractive summarizer followed by a bottom-up abstractive summarizer (Gehrmann et al., 2018). DynE is a BART-based abstractive approach, which is to ensemble multiple input, allowing single document summarization models to be directly leveraged on the multi-document summarization task. Our model outperforms all the models by a large margin, including the SOTA model DynE, and it may indicate that the plain structure is more effective than purely ensembling the output of single documents.

[15]Due to the lack of computational resources, we do not train the model on Wikisum.

| Models | R1 | R2 | RL* |
|---|---|---|---|
| LEAD | 27.46 | 4.57 | - |
| BERTABS | 31.56 | 5.02 | - |
| BART | 32.83 | 6.36 | - |
| SCIBERTABS | 32.12 | 5.59 | - |
| SOTA(Pointer Generator) | **34.11** | 6.76 | 18.2 |
| LEAD(ours) | 26.49 | 4.26 | 14.70 |
| Ours | 31.93 | **7.37** | 18.02 |

Table 10: ROUGE scores of the previous models and our fully supervised model on the Multi-Xscience dataset. All the results are from Lu et al. (2020). * The ROUGE-L is not comparable as we have different settings on the settings of evaluation, see the gap between LEAD and LEAD(ours).

| Models | R1 | R2 | RL |
|---|---|---|---|
| BERTREG (Gholipour Ghalandari et al., 2020) | 35.0 | 13.5 | 25.5 |
| SUBMODULAR+ABS(Gholipour Ghalandari et al., 2020) | 30.6 | 10.1 | 21.4 |
| DynE (Hokamp et al., 2020) | 35.4 | 15.1 | 25.6 |
| Ours | **46.08** | **25.21** | **37.86** |

Table 11: ROUGE scores of the previous models and our fully supervised model on the WCEP dataset.

| Models | R1 | R2 | RL |
|---|---|---|---|
| PEGASUS (1K) | 44.21 | 16.95 | 38.83 |
| Bigbird-PEGASUS (3k) | 46.63 | 19.02 | 41.77 |
| LED(4K) | 44.40 | 17.94 | 39.76 |
| LED(16K) | 46.63 | 19.62 | 41.83 |
| Ours(4k) | **47.58** | **20.75** | **42.57** |

Table 12: ROUGE scores of the previous models and our fully supervised model on the arXiv dataset. The result of PEGASUS and BigBird-PEGASUS are from (Zaheer et al., 2020), and the results of LED are from (Beltagy et al., 2020). The number in the parenthesis indicates the length limit of the input.

**Document #1** Wildfires have burned across tens of thousands of acres of parched terrain in Colorado, spurring thousands of evacuations ...(0.107)..., residents have sought shelter in middle schools, and local officials fear tourists usually drawn to the region for the summer may not come.

**Document #2** ... *In Colorado's southwest, authorities have shuttered the San Juan National Forest in southwestern Colorado and residents of more than 2,000 homes were forced to evacuate.(0.187)* No homes had been destroyed ... *"Under current conditions, one abandoned campfire or spark could cause a catastrophic wildfire, ..., with human life and property," said San Juan National Forest Fire Staff Officer Richard Bustamante*...

**Document #3** The Buffalo Fire west of Denver is ... Several wildfires in Colorado have prompted thousands of home evacuations ...(0.172)... Nearly 1,400 homes have been evacuated in Summit County, Colorado, ...(0.179)... "Under current conditions, one abandoned campfire or spark could cause a catastrophic wildfire, ... , with human life and property," said Richard Bustamante, SJNF forest fire staff officer ...

**Entities with High Frequency**

Colorado, 416, Tuesday, Wildfires, San Juan National Forest,...

Figure 6: An example on sentence selection by Principle vs our Entity Pyramid strategy. Italic text in red is the sentence with the highest Principle ROUGE scores, which is thereby chosen by the Principle Strategy. Most frequent entity 'Colorado' is shown with blue, followed by the Pyramid ROUGE scores in parenthesis. The final selected sentence by Entity Pyramid strategy is in *italic*. which is a better pseudo-summary than the ones selected by the Principle strategy.

**arXiv** In addition to the experiments on multi-document summarization datasets, we also compare our fully supervised model with previous works on the arXiv dataset, with each section treated as a single document. All the models to be compared with are based on pre-trained models, and Bigbird-PEGASUS and LED utilize the pre-training of PEGASUS (Zaheer et al., 2020) and BART (Lewis et al., 2020), respectively. However, both Bigbird and LED apply more efficient attentions, which make the models able to take longer input (3k for BigBird, 4K and 16k for LED). Our model has a better performance than all the models, including LED(16K), which allows for the input 4 times longer than ours. It is worth mentioning that LED(4K) has the same structure as our model, with the same length limit of the input, and with the pre-training on multi-document datasets, our model is more than 3 ROUGE point better than it, which shows that the strategy not only works for multi-document summarization but can also effectively improve single-document summarization for long documents.

## G Sentence Selection Example

Figure 6 shows an example of sentences picked by the Principle strategy (Zhang et al., 2020) vs our Entity Pyramid approach. The figure shows a cluster containing three news articles discussing a wildfire happened in Corolado, and the pseudo-summary of this cluster should be related to the location, time and consequence of the wildfire, but with the Principle strategy, the non-salient sentences quoting the words from an officer are assigned the highest score, as the exact same sentence appeared in two out of the three articles. In comparison, instead of the quoted words, our strategy selects the most representative sentences in the cluster with high frequency entities.

## H Details on Human Evaluation

In this section, we explain the details of human evaluation.

**Settings** We use 10 data examples for training, 10 data examples for validation and 20 data examples for testing for both datasets. Three annotators[16] are hired to do both Pyramid Evaluation and Fluency evaluation, they harmonize the standards on one of the examples. Specifically, for each data example, we provide three anonymized system generated summaries, along with a list of SCUs. The annotators are asked to find all the covered SCUs for each summary, and score the fluency in terms of Grammaticality, Referential clarity and Structure & Coherence, according to DUC human evaluation guidelines, with a scale 1-5 (worst to best). They are also suggested to make comparison between three generated summaries into consideration when scoring the fluency. To control for the ordering effect of the given summaries, we re-order the three summaries for each data example, and ensure the chance of their appearance in different order is the same (e.g. BART appears as summary A for 7 times, B for 7 times and C for 6 times for both datasets). The instruction for human annotation can be found in Figure 7 and Figure 8. Annotators were aware that annotations will be used solely for computing aggregate human evaluation metrics and reporting in the scientific paper.

**Pyramid Evaluation** Both TAC and DUC datasets include SCU (Summary Content Unit) annotations and weights identified by experienced annotators. We then ask 3 annotators to make a binary decision whether each SCU is covered in a candidate summary. Following Nenkova and Passonneau (2004), the raw score of each summary is then computed by the sum of weights of the covered SCUs, i.e. $S_r = \sum_{SCU} w_i I(SCU_i)$, where

[16] We recruited expert annotators with payment above average of the participants' demographics.

$I(SCU_i)$ is an indicator function on whether $SCU_i$ is covered by the current summary, and $w_i$ is the weight of $SCU_i$. In the original pyramid evaluation, the final score is computed by the ratio of $S_r$ to the maximum possible weights with the same number of SCUs as in the generated summaries. However, the total number of SCUs of generated summaries is not available in the simplified annotations in our design. To take consideration of the length of generated summaries and make a fair comparison, instead, we compute Recall, Precision and F-1 score regarding lengths of both gold references and system generated summaries as

$$\text{R}=\frac{S_r}{len(gold)};\quad \text{P}=\frac{S_r}{len(sys)};\quad \text{F1}=\frac{2 \cdot R \cdot P}{(R+P)}$$

## I Examples of Generated Summaries

We show an example (from Multi-News) of generated summaries by PRIMERA and compared models trained with different number of examples in Table 13. And we show an example from DUC2007 (which is one of the examples used for human evaluation) with generated summaries by PRIMERA and two compared models in Table 14, with all the models trained on 10 data examples from DUC2007.

## J Software and Licenses

Our code is licensed under Apache License 2.0. Our framework dependencies are:

- HuggingFace Datasets[17], Apache 2.0
- NLTK [18], Apache 2.0
- Numpy[19], BSD 3-Clause "New" or "Revised"
- Spacy[20], MIT
- Transformers[21], Apache 2.0
- Pytorch[22], Misc
- Pytorch Lightning [23],Apache 2.0
- Longformer[24], Apache 2.0
- ROUGE [25], Apache 2.0

[17] https://github.com/huggingface/datasets/blob/master/LICENSE

[18] https://github.com/nltk/nltk

[19] https://github.com/numpy/numpy/blob/main/LICENSE.txt

[20] https://github.com/explosion/spaCy/blob/master/LICENSE

[21] https://github.com/huggingface/transformers/blob/master/LICENSE

[22] https://github.com/pytorch/pytorch/blob/master/LICENSE

[23] https://github.com/PyTorchLightning/pytorch-lightning/blob/master/LICENSE

[24] https://github.com/allenai/longformer/blob/master/LICENSE

[25] https://github.com/google-research/google-research/tree/master/rouge

# Instruction on Human Annotations

Overview:

**1- Content quality**

The goal of this evaluation is to assess the content quality of a system generated summary based on human-written reference summaries. We will be using the Pyramid evaluation framework (Nenokva and Passonneau, 2004): https://aclanthology.org/N04-1019.pdf

The evaluation works like this. There are 3 things provided for each evaluation example:
1- The original documents
2- System generated summaries
3- A set of "information nuggets" or "facts" in the human-reference summaries. These are pre-annotated in the dataset and are atomic short phrases that refer to a specific piece of information in the documents. For example *"Britain opted out of Euro system."* is an information nugget.

For annotation, we will simply go through the list of information nuggets and check if they appear in the system generated summary. A summary is considered good if it includes many of the information nuggets.

1. Documents (40 examples in total)
    a. Information nuggets: https://docs.google.com/spreadsheets/d/1BRbv-aaVpNDWTKOhQBGYABYzhKWAhk6cGnD5ROlyquY/edit?usp=sharing
    b. Three summaries, A, B and C: See below
2. Annotations:
    a. **For each summary**, check all the information in the information checklist in the following way,
        i. If the information is covered in the summary, check the corresponding checkbox
        ii. Otherwise, leave it blank.

**2- Fluency:**

There are multiple candidate summaries (3) for each system and we would like to rank the summaries from most most fluent to least.

Specifically, for each summary, answer the following questions:

Figure 7: Annotation instruction for human annotators.

1. **Grammaticality** - The summary should have no datelines, system-internal formatting, capitalization errors or obviously ungrammatical sentences (e.g., fragments, missing components) that make the text difficult to read.
    5. Very Good
    4. Good
    3. Barely Acceptable
    2. Poor
    1. Very Poor

2. **Referential clarity** - It should be easy to identify who or what the pronouns and noun phrases in the summary are referring to. If a person or other entity is mentioned, it should be clear what their role in the story is. So, a reference would be unclear if an entity is referenced but its identity or relation to the story remains unclear.
    5. Very Good
    4. Good
    3. Barely Acceptable
    2. Poor
    1. Very Poor
3. **Structure and Coherence** - The summary should be well-structured and well-organized. The summary should not just be a heap of related information, but should build from sentence to sentence to a coherent body of information about a topic.
    5. Very Good
    4. Good
    3. Barely Acceptable
    2. Poor
    1. Very Poor

When assigning scores to each summary, please assign the scores relative to other summaries for that document set.

Figure 8: Annotation instruction for human annotators.

| Model | Summaries |
|---|---|
| PRIMERA-0 | BALTIMORE (AP) — The U.S. Coast Guard says it's looking for a man who has cost the service about $500,000 after responding to nearly 30 of his fake distress calls.. The Coast Guard also said such false distress alerts detract from their ability to respond to actual alerts.Anyone with information about the caller is asked to contact the Coast Guard's regional command center at 410-576-2525 or email investigators at CGIS-Baltimore@uscg.mil.. |
| PRIMERA-10 | The U.S. Coast Guard says it's looking for a man who has cost the service about $500,000 after responding to nearly 30 of his fake distress calls. In a press release published Friday, the Coast Guard says the 28 calls have originated from around the area of Annapolis, Maryland. Each call involved the same male voice and used an emergency radio channel. The two most recent calls were made on the night of July 21 and the early morning of July 22. The Coast Guard also says hoax calls distract rescuers from real emergencies, putting both the public and the responding crews at risk. "Calls like these not only put our crews at risk, but they put the lives of the public at risk," said Lt. Cmdr. Sara Wallace, who heads the response sector in Maryland. "Calls like these not only put our crews at risk, but they put the lives of the public at risk." Anyone with information about the caller is asked to contact the Coast Guard's regional command center at 410-576-2525 or email investigators at CGIS-Baltimore@uscg.mil. |
| PRIMERA-100 | The US Coast Guard is looking for a man who has cost the service about $500,000 after making 28 false distress calls from Maryland. The 28 calls have involved the same male voice and used an emergency radio channel, the AP reports. The Coast Guard says it has responded to nearly 30 of the fake calls and that the estimated cost of the responses is $500,000. The caller has been making the calls since July 2014. He could be facing a six-year prison sentence, a $10,000 civil fine, and a $250,000 criminal fine, the AP reports. "A hoax call is a deadly and serious offense," says Lt. Cmdr. Sara Wallace, head of the response sector in Maryland. "Calls like these not only put our crews at risk, but they put the lives of the public at risk." |
| PEGASUS-100 | The US Coast Guard has responded to 28 false distress calls in two years—and it's a man who's been making the calls, not the waves. The unidentified caller has made the calls in Annapolis, Maryland, since July 2014, the Coast Guard reports. Each call involved the same male voice and used an emergency radio channel. The two most recent calls were made on the night of July 21 and early morning of July 22. The Coast Guard says the calls were made in the hope that the public would get into the water. Rather than responding to actual alerts, the calls were made to divert the attention of rescue crews from real emergencies, the AP reports. The Coast Guard says the cost of the responses to these false alarms is about $500,000. The caller made 28 false distress alerts from Annapolis, Maryland, beginning in July 2014, according to the Coast Guard. The estimated cost of the responses to these false alarms is $500,000. The Coast Guard says the calls were made in the hope that the public would get into the water. Rather than responding to actual alerts, the calls were made to divert the attention of rescue crews from real emergencies, the AP reports. |
| Fully Supervised | The US Coast Guard says it's looking for a man who has cost the service about $500,000 after responding to nearly 30 of his fake distress calls, the AP reports. In a press release published Friday, the Coast Guard says the 28 calls have originated from around the area of Annapolis, Maryland. Each call involved the same male voice and used an emergency radio channel. The two most recent calls were made on the night of July 21 and the early morning of July 22. The Coast Guard also says hoax calls distract rescuers from real emergencies, putting both the public and the responding crews at risk. "A hoax call is a deadly and serious offense," says Lt. Cmdr. Sara Wallace, who heads the response sector in Maryland. "Calls like these not only put our crews at risk, but they put the lives of the public at risk." Making false distress calls is a felony, punishable by a maximum of six years in prison, a $10,000 civil fine, $250,000 criminal fine, and a reimbursement to the Coast Guard for their efforts, the Coast Guard says. |
| Ground Truth | The US Coast Guard says it's looking for a man who has cost the service about $500,000 after responding to nearly 30 of his fake distress calls, reports the AP. In a press release published Friday, the Coast Guard says the 28 calls have originated from around the area of Annapolis, Maryland. Each call involved the same male voice and used an emergency radio channel. He's been making the calls since July 2014. The two most recent calls were made on the night of July 21 and the early morning of July 22. "A hoax call is a deadly and serious offense," a Coast Guard rep tells NBC4, which notes that such calls are a felony that carry six years in prison, $10,000 civil fine, $250,000 criminal fine, and reimbursement to the Coast Guard. "Calls like these not only put our crews at risk, but they put the lives of the public at risk." |

Table 13: Generated summaries from PRIMERA and best baseline model (according ROUGE score on this example) trained with different number of training examples. The data used here is the #10 in the test set of Multi-News dataset on Huggingface.

| Model | Summaries |
|---|---|
| PEGASUS | In 1996, Congress passed the Line-Item Veto Act, which gave the president the power to cut individual projects from tax and spending bills without vetoing the entire legislation. The act was followed by the President's line-item veto, which he used to trim 144 million dollars from a 248 billion dollars defense spending bill. He also used the veto power to block a congressional rejection of his line-item veto on 38 military construction projects. The bill was passed by the House and the President signed it into law. The veto was challenged by members of both parties who said it was unconstitutional because it gave the president unchecked power to rewrite legislation. The Supreme Court agreed on Friday to hear argument and decide the constitutionality of the president line-item veto. In 1998 the President used his line-item veto to cut $38 million from a military construction bill. In 1999 the President used his line-item veto to cut $54 million from a military spending bill. In 2000 the President used his line-item veto to cut $54 million from a defense spending bill. In January the President vetoed a tax and spending bill, which gave him the power to cut individual projects from tax and spending bills without vetoing the entire legislation. In February the President vetoed a spending bill, which gave him the power to cut individual projects from tax and spending bills without vetoing the entire legislation. In September the President used his line-item veto to cut $54 million from a defense spending bill. The bill was rejected by the House and the President vetoed it. In November the President used his line-item veto to trim 144 million dollars from a defense spending bill. |
| LED | In 1996, the Republican-led Congress passed the Line Item Veto Act, giving the president the power to delete individual items of spending and tax bills. Clinton used the power to cut individual projects from tax and spending bills. In February 1999, the President Clinton vetoed a congressional rejection of his line-item veto on 38 military construction projects. In May 1999, the President Clinton used the line-item veto to cut individual items of spending and tax breaks. In 2000, the President Clinton used the line-item veto to cancel individual items of spending and tax breaks. In May 2000, the President Clinton threatened to use the line-item veto to cancel all military spending and tax breaks. In June 2000, the President Clinton used the line-item veto to cut individual items of spending and tax breaks. In August 2000, the President Clinton used the line-item veto to cut individual items of spending and tax breaks. In September 2000, the President Clinton used the line-item veto to cut individual items of spending and tax breaks. In 2000, the President Clinton used the line-item veto to cut individual items of spending and tax breaks. In 2001, the President Clinton used the line-item veto to cut individual items of spending and tax breaks. In June 2000, the President Clinton used the line-item veto to cut individual items of spending and tax breaks. In August 2000, the President Clinton used the line-item veto to cut individual items of spending and tax breaks. In September 2000, the President Clinton used the line-item veto to cut individual items of spending and tax breaks. In 2000, the President Clinton used the line-item veto to cancel individual items of spending and tax breaks. In 2001, the President Clinton used the line-item veto to cut individual items of spending and tax breaks. In June 2000, the President Clinton used the line-item veto to cut individual items of spending and tax breaks. |
| PRIMERA | In 1996, Congress gave the President the power to veto specific items of tax and spending bills. Before the law's enactment in 1996, the president could veto an entire spending measure but not individual items. The court ruled that such a specialized veto can be authorized only through a constitutional amendment. In January 1997, the line-item veto law was passed. It was passed under the Republican Party's "Contract with Congress". It was passed after President Clinton vetoed thirteen relatively obscure research and spending programs, almost all of the military spending increases approved by Congress. In October 1998, Clinton used his line-item veto authority to have trimmed 144 million U.S. dollars from a 248 billion defense spending bill. In November 1998, Clinton vetoed 38 military construction projects, worth 287 million U.S. dollars. In February 1999, the Justice Department appealed the line-item veto law to the Supreme Court, which agreed to hear argument and decide the constitutionality of the law. Earlier this month, a federal judge struck down the line-item veto law as unconstitutional. The highest court's review will yield a momentous balance of powers ruling. The case is scheduled to be argued before the justices on April 27. The line item veto, strongly supported by President Bill Clinton and a number of his predecessors, was passed in 1996 under the Republican Party's "Contract with Congress". It was passed in January 1997. Before the law's enactment, the only way presidents could reject spending laws was to veto whole budget bills. In 1996, Congress gave the president the power to cancel individual items in tax and spending bills. In January 1997, the line-item veto law was passed. It was passed under the Republican Party's "Contract with Congress". It was passed in January 1997. In 1998, President Clinton threatened to veto some items of the military construction bill because of the increased funding. In November 1998, Clinton used his line-item veto power to delete 38 projects in 24 states worth 287 million U.S. dollars. In February 1999, the Justice Department appealed the line-item veto law to the Supreme Court, which agreed to hear a case about its constitutionality. |
| Ground Truth | In 1996 a Republican congress overwhelmingly passed a Line Item Veto Act allowing presidents (including the incumbent Democratic president), to strike individual tax or spending items within 5 days after signing a bill into law. Congress could restore those items in a new bill passed by majority vote. If the president vetoed that bill, Congress could override that veto with a two-thirds majority. Proponents argued that the law preserved the integrity of federal spending, saved billions of dollars, and that it did not repeal any portion of a law, but was simply a delegated spending authorization from Congress. In January 1997, the first year of the law, the president vetoed 163 line-items in six bills, and in 1998 82 line-items in 11 bills. In October 1997 Congress overrode the president's line-item veto against 36 of 38 military construction projects. Initial 1997 efforts by congressmen to challenge the law in the Supreme Court were rejected due to lack of standing. On June 25, 1998 after lower courts rejected the Line Item Veto Act as unconstitutional, on appeal by the White House the Supreme Court ruled 6-3 that Congress unconstitutionally violated the principle of separation of powers, because that procedure allows the president to create a law that was not voted on by either house of Congress in violation of the Constitution's Article I "presentment" clause. A constitutional amendment would be required to institute line item vetoes. Justices Breyer and Scalia argued similar dissenting opinions that separation of powers was not violated. |

Table 14: Generated summaries from PRIMERA, PEGASUS and LED trained with 10 training examples, along with one (out of four) ground-truth summary. The data used here is D0730 in DUC2007.