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

Figure 1: PRIMERA vs existing pretrained models.

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

\[\text{sent-mask}\]

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 which was originally developed for evaluating summarization with respect to the cluster of related documents. Algorithm 1 shows the sentence selection procedure.

---

**Algorithm 1 Entity Pyramid Sentence Selection**

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Input: Document cluster</td>
</tr>
<tr>
<td>2.</td>
<td>Input: List of entities w/ frequency &gt; 1. N length of the list</td>
</tr>
<tr>
<td>3.</td>
<td>Input: m number of sentences to select</td>
</tr>
<tr>
<td>4.</td>
<td>Output: List of sentences to mask</td>
</tr>
</tbody>
</table>

1. \( E \leftarrow \) sort entities by frequency, descending
2. \( selected \leftarrow [\] \)
3. \( \text{for } i \leftarrow 1 \text{ to } |E| \text{ do} \)
4. \( \text{SentCand} \leftarrow \) all sentences in the cluster containing \( E[s] \)
5. \( \text{cur_sent} = \arg \max_{s \in \text{SentCand}} \text{Score}(s) \)
6. \( \text{selected}.\text{append}(\text{cur_sent}) \)
7. \( \text{if } [\text{selected}] == m \text{ then} \)
8. \( \text{Break} \)
9. \( \text{end if} \)
10. \( \text{end for} \)
11. Return \( \text{selected} \)

---

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 highest to lowest frequency. For large scale pretraining, as a proxy, we explores 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:

---

3Note 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.
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

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.
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.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R-1</td>
<td>R-2</td>
<td>R-L</td>
<td>R-1</td>
<td>R-2</td>
<td>R-L</td>
</tr>
<tr>
<td>PEGASUS+ (Zhang et al., 2020)</td>
<td>36.5</td>
<td>10.5</td>
<td>18.7</td>
<td>28.1</td>
<td>6.6</td>
<td>17.7</td>
</tr>
<tr>
<td>PEGASUS (our run)</td>
<td>32.0</td>
<td>10.1</td>
<td>16.7</td>
<td>27.6</td>
<td>4.6</td>
<td>15.3</td>
</tr>
<tr>
<td>BART (our run)</td>
<td>27.3</td>
<td>6.2</td>
<td>15.1</td>
<td>18.9</td>
<td>2.6</td>
<td>12.3</td>
</tr>
<tr>
<td>EGASUS (Zhang et al., 2020)</td>
<td>⋆</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PODER (our model)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRIMERA (our model)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For clarity, by zero-shot we mean using the pretrained models directly without any additional supervision.
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. 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 pretrained 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, 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...

---

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

10Due to the lack of computational resources, we do not train the model on Wikisum.
Table 3: Fully supervised results. Previous SOTA are from Pasunuru et al. (2021) for Multi-News, Lu et al. (2020) for Multi-XScience, Hokamp et al. (2020) for WCEP, and Beltagy et al. (2020) for arXiv.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Previous SOTA</th>
<th>PRIMERA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R-1</td>
<td>R-2</td>
</tr>
<tr>
<td>Multi-News</td>
<td>49.2</td>
<td>19.6</td>
</tr>
<tr>
<td>Multi-XScience</td>
<td>33.9</td>
<td>6.8</td>
</tr>
<tr>
<td>WCEP</td>
<td>35.4</td>
<td>15.1</td>
</tr>
<tr>
<td>arXiv</td>
<td>46.6</td>
<td>19.6</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>Model</th>
<th>DUC2007(20)</th>
<th>TAC2008(20)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$S_r$</td>
<td>$R$</td>
</tr>
<tr>
<td>PEGASUS</td>
<td>6.0</td>
<td>2.5</td>
</tr>
<tr>
<td>LED</td>
<td>9.6</td>
<td>3.9</td>
</tr>
<tr>
<td>PRIMERA</td>
<td>12.5</td>
<td>5.1</td>
</tr>
</tbody>
</table>

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

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 (\langle\text{doc-sep}\rangle 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 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. 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. $S_r = \sum_{SCU} w_i I(\text{SCU}_i)$, where $I(\text{SCU}_i)$ is an indicator function on whether $\text{SCU}_i$ is covered by the current summary, and $w_i$ is the

---

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

<table>
<thead>
<tr>
<th>Model</th>
<th>DUC2007(20)</th>
<th>TAC2008(20)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PEGASUS</td>
<td>4.45</td>
<td>4.35</td>
</tr>
<tr>
<td>LED</td>
<td>4.35</td>
<td>4.50</td>
</tr>
<tr>
<td>PRIMERA</td>
<td>4.70</td>
<td>4.65</td>
</tr>
</tbody>
</table>

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.

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


A Implementation details of pre-training

As the multi-document summarization task has a higher compression ratio, defined as \( \text{len}(\text{Summary})/\text{len}(\text{Input}) \), (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.\(^\text{14}\) 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 Huguenews 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 ± 0.79 vs. (24.13) and 25.86 ± 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

\(^\text{14}\) 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.
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 Hyperparameters in Few-shot and Full Supervised Experiments

### D.1 Few-shot Experiments

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.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.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Total Steps</th>
<th>Warmup Steps</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-News</td>
<td>25k</td>
<td>2.5k</td>
</tr>
<tr>
<td>Multi-XScience</td>
<td>20k</td>
<td>2k</td>
</tr>
<tr>
<td>WCEP</td>
<td>5k</td>
<td>.5k</td>
</tr>
<tr>
<td>arXiv</td>
<td>40k</td>
<td>4k</td>
</tr>
</tbody>
</table>

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.

### E.1 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.
can be found in Table 9-12\textsuperscript{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).

**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.

\textsuperscript{15}Due to the lack of computational resources, we do not train the model on Wikisum.
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 ... (0.179)... Nearly 1,400 homes have been evacuated in Summit County, Colorado, ... “Under current conditions, one abandoned campfire or spark could cause a catastrophic wildfire, ... with human life and property,” said Richard Bustamante, SINF 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 Colorado, 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.

II 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 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 Pasca (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

16We 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

$$R = \frac{S_r}{\text{len}(\text{gold})}; \quad P = \frac{S_r}{\text{len}(\text{sys})}; \quad F_1 = \frac{2 \cdot R \cdot P}{(R + P)}$$

### 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

\(^{17}\)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/1BRby-aaVpNDWTKOhQBGYABYzhKWAhk6cGn5ROlyquY/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.
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 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.

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.
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’s 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 February 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.

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, Clinton used the line-item veto to cancel individual items of spending and tax breaks. In 2000, the President Clinton used 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 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 2001, 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 2001, 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 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 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 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 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 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 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 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 2000, the President Clinton used the line-item veto to cut individual items of spending and tax breaks."