Recent work pre-training Transformers with self-supervised objectives on large text corpora has shown great success when fine-tuned on downstream NLP tasks including text summarization. However, pre-training objectives tailored for abstractive text summarization have not been explored. Furthermore there is a lack of systematic evaluation across diverse domains. In this work, we propose pre-training large Transformer-based encoder-decoder models on massive text corpora with a new self-supervised objective. In PEGASUS, important sentences are removed/masked from an input document and are generated together as one output sequence from the remaining sentences, similar to an extractive summary. We evaluated our best PEGASUS model on 12 downstream summarization tasks spanning news, science, stories, instructions, emails, patents, and legislative bills. Experiments demonstrate it achieves state-of-the-art performance on all 12 downstream datasets measured by ROUGE scores. Our model also shows surprising performance on low-resource summarization, surpassing previous state-of-the-art results on 6 datasets with only 1000 examples. Finally we validated our results using human evaluation and show that our model summaries achieve human performance on multiple datasets.

1 Introduction

Text summarization aims at generating accurate and concise summaries from input document(s). In contrast to extractive summarization which merely copies informative fragments from the input, abstractive summarization may generate novel words. A good abstractive summary covers principal information in the input and is linguistically fluent.

In abstractive summarization, sequence-to-sequence (Sutskever et al., 2014) has become a dominant framework using encoder-decoder architectures based on RNNs (Chung et al., 2014; Hochreiter & Schmidhuber, 1997) and more recently Transformers (Vaswani et al., 2017). Most prior work on neural abstractive summarization relied on large-scale, high-quality datasets of supervised document-summary pairs (Hermann et al., 2015) and achieved promising results (Rush et al., 2015; Nallapati et al., 2016; See et al., 2017). In recent years, there has been increased interest in collecting new summarization datasets that have more abstractive summaries (Narayan et al., 2018), have longer documents, (Cohan et al., 2018; Sharma et al., 2019), utilize multiple documents (Fabbri et al., 2019), and are sourced from diverse domains (Grusky et al., 2018; Kourpae & Wang, 2018; Kim et al., 2019; Kornilova & Eidelman, 2019; Zhang & Tetreault, 2019);
however, there has been little work on systematic evaluation of models across these broad settings.

Contemporaneously, the adoption of Transformer models (Vaswani et al., 2017) pre-trained using self-supervised objectives on large text corpora (Radford et al., 2018a; Devlin et al., 2019) have improved performance on many NLP tasks (Wang et al., 2018; Rajpurkar et al., 2016).

Recent work leveraging such pre-training for Transformer-based sequence-to-sequence models (Dong et al., 2019; Song et al., 2019; Rothe et al., 2019; Lewis et al., 2019; Raffel et al., 2019) has extended the success to text generation, including abstractive summarization.

In this work, we study pre-training objectives specifically for abstractive text summarization and evaluate on 12 downstream datasets spanning news (Hermann et al., 2015; Narayanan et al., 2018; Grusky et al., 2018; Rush et al., 2015; Fabri et al., 2019), science (Cohan et al., 2018), short stories (Kim et al., 2019), instructions (Koupaee & Wang, 2018), emails (Zhang & Tetreault, 2019), patents (Sharma et al., 2019), and legislative bills (Kornilova & Eidelman, 2019). We find that masking whole sentences from a document and generating these gap-sentences from the rest of the document works well as a pre-training objective for downstream summarization tasks. In particular, choosing putatively important sentences outperforms lead or randomly selected ones. We hypothesize this objective is suitable for abstractive summarization as it closely resembles the downstream task, encouraging whole-document understanding and summary-like generation. We call this self-supervised objective Gap Sentences Generation (GSG). Using GSG to pre-train a Transformer encoder-decoder on large corpora of documents (Web and news articles) results in our method, Pre-training With Extracted Gap-sentences for Abstractive Summarization Sequence-to-sequence models, or PEGASUS.

With our best 568M parameter model trained on the recently introduced C4 (Raffel et al., 2019) corpus we equal or exceed state-of-the-art on the 12 summarization tasks we consider. We further push forward the state-of-the-art using a newly collected text corpus comprised of news-like articles we call HugeNews, including the highly competitive XSSum and CNN/DailyMail summarization datasets.

Large-scale document-summary datasets are rare and in practice there is a mismatch between research datasets and real-world use-cases where collecting summaries is expensive; the most common setting is that of low-resource summarization. We simulate this setting and show that our model is able to adapt very quickly when fine-tuning with small numbers of supervised pairs, obtaining state-of-the-art results in 6 datasets with only 1000 examples.

Qualitatively we observed high quality outputs from our best models and validated this in human evaluation studies. We found that PEGASUS summaries are at least as good as reference summaries for the datasets we assessed – XSSum, CNN/DailyMail, and Reddit TIFU – even at low-levels of supervision.

To summarize our contributions:

- We propose a new self-supervised pre-training objective for abstractive summarization, gap-sentences generation, and study strategies for selecting those sentences.
- We evaluate the proposed pre-training objective on a broad range of downstream summarization tasks, with careful ablations to choose the best model settings, which we use to train a 568M parameter PEGASUS model that surpasses or is on-par with the state-of-the-art on all 12 downstream datasets considered.
- We show how good abstractive summarization performance can be achieved across broad domains with very little supervision by fine-tuning the PEGASUS model and surpassing previous state-of-the-art results on many tasks with as little as 1000 examples.
- We conducted human evaluation studies to validate our experimental design and demonstrate human-level summarization performance on XSSum, CNN/DailyMail, and Reddit TIFU.

2 Related Work

Dai & Le (2015); Ramachandran et al. (2017) used LM and autoencoder pre-training on in-domain data to improve performance of RNN sequence models. However, the combination of pre-training with much larger external text corpora (such as Wikipedia, books, or Web-pages) and Transformer-based sequence models has led to a dramatic improvement in performance when fine-tuned for both natural language understanding and text generation tasks (Radford et al., 2018a; Devlin et al., 2019; Rothe et al., 2019; Yang et al., 2019; Joshi et al., 2019; Song et al., 2019; Dong et al., 2019; Lewis et al., 2019). Most similar to our approach are Transformer encoder-decoder models pre-trained on some masked input pre-training objective.

MASS (Song et al., 2019) proposed masked sequence-to-sequence generation that reconstructs a sentence fragment given the remaining part of the sentence. A single sentence fragment was randomly selected.

UniLM (Dong et al., 2019) proposed jointly training on three types of language modeling tasks: unidirectional (left-to-right and right-to-left), bidirectional (word-level mask,
with next sentence prediction), and sequence-to-sequence (word-level mask) prediction.

**T5** (Raffel et al., 2019) generalized the text-to-text framework to a variety of NLP tasks and showed the advantage of scaling up model size (to 11 billion parameters) and pre-training corpus, introducing C4, a massive text corpus derived from Common Crawl, which we also use in some of our models. T5 was pre-trained with randomly corrupted text spans of varying mask ratios and sizes of spans.

**BART** (Lewis et al., 2019) introduced a denoising autoencoder to pre-train sequence-to-sequence models. BART corrupted text with an arbitrary noising function and learned to reconstruct the original text. For generation tasks, the noising function was text infilling which used single mask tokens to mask random sampled spans of text.

In contrast to MASS, UniLM, BART and T5, the proposed PEGASUS masks multiple whole sentences rather than smaller continuous text spans. In our final objective we deterministically choose sentences based on importance, rather than randomly. As in T5, PEGASUS does not reconstruct full input sequences, and only generates the masked sentences as a single output sequence. In this work we focus entirely on downstream summarization (generative) tasks and do not evaluate on NLU classification tasks.

There has been some work on the low-resource, summarization setting using the CNN/DailyMail dataset. Radford et al. (2018b) showed that a large Transformer language model pre-trained on Web text could generate summaries if prompted with “TL;DR”, achieving a ROUGE-2 of 8.27 on CNN/DailyMail. Khandelwal et al. (2019) pre-trained a Transformer language model on Wikipedia, and fine-tuned using 3000 examples, achieving 13.1 ROUGE-2.

### 3 Pre-training Objectives

We propose a new pre-training objective, GSG, in this work, but for comparison, we also evaluate BERT’s masked-language model objective, in isolation and in conjunction with GSG.

#### 3.1 Gap Sentences Generation (GSG)

We hypothesize that using a pre-training objective that more closely resembles the downstream task leads to better and faster fine-tuning performance. Given our intended use for abstractive summarization, our proposed pre-training objective involves generating summary-like text from an input document. In order to leverage massive text corpora for pre-training, we design a sequence-to-sequence self-supervised objective in the absence of abstractive summaries. A naive option would be to pre-train as an extractive summarizer; however, such a procedure would only train a model to copy sentences, thus not suitable for abstractive summarization.

Inspired by recent success in masking words and contiguous spans (Joshi et al., 2019; Raffel et al., 2019), we select and mask whole sentences from documents, and concatenate the gap-sentences into a pseudo-summary. The corresponding position of each selected gap sentence is replaced by a mask token [MASK1] to inform the model. Gap sentences ratio, or GSR, refers to the number of selected gap sentences to the total number of sentences in the document, which is similar to mask rate in other works.

To even more closely approximate a summary, we select sentences that appear to be important/principal to the document. The resulting objective has both the empirically demonstrated benefits of masking, and anticipates the form of the downstream task.

We consider 3 primary strategies for selecting m gap sentences without replacement from a document, $D = \{x_i\}_n$, comprised of n sentences:

- **Random** Uniformly select m sentences at random.
- **Lead** Select the first m sentences.
- **Principal** Select top-m scored sentences according to importance. As a proxy for importance we compute ROUGE1-F1 (Lin, 2004) between the sentence and the rest of the document, $s_i = \text{rouge}(x_i, D \setminus \{x_i\}), \forall i$.

In this formulation sentences are scored independently (Ind) and the top m selected. We also consider selecting them sequentially (Seq) as in Nallapati et al. (2017) by greedily maximizing the ROUGE1-F1 between selected sentences, $S \cup \{x_i\}$, and remaining sentences, $D \setminus (S \cup \{x_i\})$ as in Algorithm 1.

**Algorithm 1 Sequential Sentence Selection**

1: $S := \emptyset$
2: for $j \leftarrow 1$ to $m$ do
3:     $s_i := \text{rouge}(S \cup \{x_i\}, D \setminus (S \cup \{x_i\}))$
4:     $\forall i \text{ s.t. } x_i \notin S$
5:     $k := \text{arg max}_i \{s_i\}_n$
6:     $S := S \cup \{x_k\}_n$
7: end for

When calculating ROUGE1-F1, we also consider n-grams as a set (Uniq) instead of double-counting identical n-grams as in the original implementation (Orig). This results in four variants of the principal sentence selection strategy, choosing Ind/Seq and Orig/Uniq options.

An example containing lead, random and principal gap sentence selection strategies are shown in Figure 2.
INVITATION ONLY We are very excited to be co-hosting a major drinks reception with our friends at Progress. This event will sell out, so make sure to register at the link above. Speakers include Rajesh Agrawal, the London Deputy Mayor for Business, Alison McGovern, the Chair of Progress, and Seema Malhotra MP. Huge thanks to the our friends at the ACCA, who have supported this event. The Labour Business Fringe at this year’s Labour Annual Conference is being co-sponsored by Labour in the City and the Industry Forum. Speakers include John McDonnell, Shadow Chancellor, and Rebecca Long-Bailey, the Shadow Chief Secretary to the Treasury, and our own Chair, Kitty Ussher. Attendance is free, and refreshments will be provided.

Figure 2: An example of sentences (from the C4 corpus) selected by Random, Lead and Ind-Orig respectively. Best viewed in color.

3.2 Masked Language Model (MLM)

Following BERT, we select 15% tokens in the input text, and the selected tokens are (1) 80% of time replaced by a mask token \[\text{[MASK2]}\], or (2) 10% of time replaced by a random token, or (3) 10% of time unchanged. We apply MLM to train the Transformer encoder as the sole pre-training objective or along with GSG. When MLM is the sole pre-training objective, the Transformer decoder shares all parameters with encoder when fine-tuning on downstream tasks following Rothe et al. (2019).

Figure 1 simultaneously shows how both GSG and MLM are applied to the same example when used in conjunction. However, we found that MLM does not improve downstream tasks at large number of pre-training steps (section 6.1.2), and chose not to include MLM in the final model PEGASUS\_LARGE (section 6.2).

4 Pre-training Corpus

For pre-training we considered two large text corpora:

- **C4**, or the Colossal and Cleaned version of Common Crawl, introduced in Raffel et al. (2019); consists of text from 350M Web-pages (750GB).

- **HugeNews**, a dataset of 1.5B articles (3.8TB) collected from news and news-like websites from 2013-2019. A whitelist of domains ranging from high-quality news publishers to lower-quality sites such as high-school newspapers, and blogs was curated and used to seed a web-crawler. Heuristics were used to identify news-like articles, and only the main article text was extracted as plain text.

5 Downstream Tasks/Datasets

For downstream summarization, we only used public abstractive summarization datasets, and access them through TensorFlow Summarization Datasets [1] which provides publicly reproducible code for dataset processing and train/validation/test splits. We used train/validation/test ratio of 80/10/10 if no split was provided, and 10% train split as validation if there was no validation split.

**XSum** (Narayan et al., 2018) consists of 227k BBC articles from 2010 to 2017 covering a wide variety of subjects along with professionally written single-sentence summaries.

**CNN/DailyMail** (Hermann et al., 2015) dataset contains 93k articles from the CNN, and 220k articles the Daily Mail newspapers. Both publishers supplement their articles with bullet point summaries. We use the non-anonymized variant used in See et al. (2017).

**NEWSROOM** (Grusky et al., 2018) is a large dataset containing 1.3M article-summary pairs written by authors and editors in the newsrooms of 38 major publications between 1998 and 2017.

**Multi-News** (Fabbri et al., 2019) is a multi-document summarization dataset consisting of 56k pairs of news articles and their human-written summaries from the site newser.com.

**Gigaword** (Rush et al., 2015) contains 4M examples extracted from news articles (seven publishers) from the Gigaword corpus (Graff et al., 2003). The task is to generate the headline from the first sentence.

**arXiv, PubMed** (Cohan et al., 2018) are two long document datasets of scientific publications from arXiv.org (113k) and PubMed (215k). The task is to generate the abstract from the paper body.

**BIGPATENT** (Sharma et al., 2019) consists of 1.3 million U.S. patents along with human summaries under nine patent classification categories.

**WikiHow** (Koupaee & Wang, 2018) is a large-scale dataset of instructions from the online WikiHow.com website. Each of 200k examples consists of multiple instruction-step paragraphs along with a summarizing sentence. The task is to generate the concatenated summary-sentences from the paragraphs.

**Reddit TIFU** (Kim et al., 2019) contains 120K posts of informal stories from the online discussion forum Reddit, more specifically the TIFU sub-reddit from 2013-Jan to 2018-Mar. The sub-reddit posts strictly follow the rule of writing a descriptive "TL;DR" summary and has higher qual-
ity than \cite{volsk2017} (which used more subreddits) based on our manual inspection. We use the TIFU-long subset (using TLDR as summaries) in the work.

**AESLC** \cite{zhang19} consists of 18k email bodies and their subjects from the Enron corpus \cite{klimt2004}, a collection of email messages of employees in the Enron Corporation.

**BillSum** \cite{kornilova2019} contains 23k US Congressional bills and human-written reference summaries from the 103rd-115th (1993-2018) sessions of Congress. We do not use the California test set which is out-of-distribution.

Following \cite{grusky2017}, the number of examples and extractive fragment coverage/density for all downstream datasets is illustrated in Appendix A.

## 6 Experiments

In a similar strategy to \cite{raffel2019}, to save time and computation we conducted pre-training ablation experiments using a reduced-size model with 223M parameters, \textsc{PEGASUS}_{\text{BASE}}, smaller batch size, and only 4 of 12 datasets before scaling up pre-training with the best settings to the final 568M parameters, \textsc{PEGASUS}_{\text{LARGE}}. The datasets (XSum, CNN/DailyMail, WikiHow and Reddit TIFU) were chosen for diversity in abstractiveness, writing style, and size.

\textsc{PEGASUS}_{\text{BASE}} had \(L = 12, H = 768, F = 3072, A = 12\) and \textsc{PEGASUS}_{\text{LARGE}} had \(L = 16, H = 1024, F = 4096, A = 16\), where \(L\) denotes the number of layers for encoder and decoder (i.e. Transformer blocks), \(H\) for the hidden size, \(F\) for the feed-forward layer size and \(A\) for the number of self-attention heads. We pre-trained \textsc{PEGASUS}_{\text{BASE}} with a batch size of 256 and \textsc{PEGASUS}_{\text{LARGE}} with a batch size of 8192. We refer to \textsc{PEGASUS}_{\text{BASE}} without pre-training as \textsc{Transformer}_{\text{BASE}}.

We used sinusoidal positional encoding following \cite{vaswani2017}. For optimization, both pre-training and fine-tuning used Adafactor \cite{shazeer2018} with square root learning rate decay and dropout rate of 0.1.

We used greedy-decoding for studies in Section 6.1 and used beam-search with a length-penalty, \(\alpha\), as in \cite{wu2016} for the final large model.

All experiments’ hyper parameters can be found in Appendix C and reported numbers are in Appendix D and E.

### 6.1 Ablations on \textsc{PEGASUS}_{\text{BASE}}

We used \textsc{PEGASUS}_{\text{BASE}} to evaluate choices of pre-training corpus, pre-training objective, and vocabulary size. For reproducibility, we evaluated the latter two using the publicly available C4 corpus.

Note that the y-axis in Figures 3, 4, 5 are normalized by the left-most bar using \(\frac{1}{3} (R_1^{\text{BASE}} + R_2^{\text{BASE}} + R_L^{\text{BASE}})\) where \(R_1, R_2, R_L\) are ROUGE F1 scores and \(R_1^{\text{BASE}}, R_2^{\text{BASE}}, R_L^{\text{BASE}}\) are the scores of the configuration corresponding to the first bar.

With more pre-training steps, the model observed more documents in the pre-training corpus. A \textsc{PEGASUS}_{\text{BASE}} model trained for 500k (highest we tried) steps did not observe all training examples on C4 nor HugeNews. Appendix B shows the number of pre-training steps had an unsurprisingly positive impact on downstream dataset performance. We used 500k steps for the ablation studies and the large model.

#### 6.1.1 Pre-training Corpus

![Figure 3: Effect of pre-training corpus.](image)

Figure 3 shows that pre-training on HugeNews was more effective than C4 on the two news downstream datasets, while the non-news informal datasets (WikiHow and Reddit TIFU) prefer the pre-training on C4. This suggests pre-training models transfer more effectively to downstream tasks when their domains are aligned better.

#### 6.1.2 Effect of Pre-training Objectives

**GSG** We compared six variants of GSG (Lead, Random, Ind-Orig, Ind-Uniq, Seq-Uniq, Seq-Uniq) while choosing 30% sentences as gap sentences. As shown in Figure 4a, Ind-Orig achieved the best performance followed by Seq-Uniq, Ind-Orig and Seq-Uniq were consistently better (or similar) than Random and Lead across the four downstream datasets. Lead had decent performance on the two news datasets but was significantly worse on the two non-news datasets, which agrees findings of lead bias in news datasets \cite{zhong2019} (which used more subreddits) than Random and Lead across the four downstream datasets.

Lead had decent performance on the two news datasets but was significantly worse on the two non-news datasets, which agrees findings of lead bias in news datasets \cite{zhong2019}. The results suggest choosing principal sentences works best for downstream summarization tasks, and we chose Ind-Orig for the \textsc{PEGASUS}_{\text{LARGE}}.

A significant hyper-parameter in GSG is the gap-sentences ratio (GSR). A low GSR makes the pre-training less challenging and computationally efficient. On the other hand, choosing gap sentences at a high GSR loses contextual in-
formation necessary to guide the generation. We compared GSRs from 15% to 75%. For a fair comparison, the original documents were truncated to have up to 400 words. The maximum input length, $L_{\text{input}}$ in the encoder and the maximum target length, $L_{\text{target}}$ in the decoder were set as 512 tokens.

Figure 4b shows that different downstream datasets had slightly different optima. The best performance always had GSR lower than 50%. The model with 15% gap sentences achieved the highest ROUGE scores on CNN/DailyMail, while XSum/Reddit TIFU and WikiHow did better with 30% and 45% respectively. When scaling up to PEGASUS$_{\text{LARGE}}$ (Section 6.2), we chose an effective GSR of 30%.

MLM As mentioned, the MLM objective can either be applied solely or together with GSG. We jointly trained MLM with GSG Ind-Orig (MLM & Ind-Orig), which masks 30% sentences and extra 15% tokens in unselected sentences, as shown in Figure 1. Figure 4a shows that the model pre-trained with MLM alone performed significantly worse and MLM & Ind-Orig had similar performance as Random. Interestingly, when comparing MLM & Ind-Orig to Ind-Orig, we empirically observed MLM improved fine-tuning performance at early pre-training checkpoints (100k - 200k steps), but inhibited further gains with more pre-training steps (500k). Therefore, we chose not to include MLM in PEGASUS$_{\text{LARGE}}$.

6.1.3 Effect of Vocabulary

We compared two tokenization methods\footnote{Implemented in https://github.com/google/sentencepiece} Byte-pair-encoding algorithm (BPE) [Wu et al. 2016; Sennrich et al. 2016], and SentencePiece Unigram algorithm (Unigram) proposed in Kudo (2018). We evaluated Unigram with different vocabulary sizes ranging from 32k to 256k. In these experiments, models were pre-trained for 500k steps on the C4 corpus with the Ind-Orig objective and 15% GSR. As shown in Figure 5, BPE and Unigram were comparable on news datasets while Unigram outperformed BPE on non-news datasets, especially WikiHow. On XSum and CNN/DailyMail, Unigram 96k achieved the highest ROUGE scores. On WikiHow and Reddit TIFU, the best configurations were Unigram 128k and 64k respectively. Therefore, we used the overall best vocabulary option Unigram 96k in PEGASUS$_{\text{LARGE}}$.

6.2 Larger Model Results

Compared with PEGASUS$_{\text{BASE}}$, the large model PEGASUS$_{\text{LARGE}}$ had increased capacity from larger hidden size ($H : 768 \rightarrow 1024$, $F : 3072 \rightarrow 4096$, $A : 12 \rightarrow 16$), number of layers ($L : 12 \rightarrow 16$) and traversed much more data, due to larger batch size ($B : 256 \rightarrow 8192$) (same number of pre-training steps, 500k). We adopted the best practices found in the PEGASUS$_{\text{BASE}}$ ablation studies using the GSG (Ind-Orig) pre-training objective without MLM and Unigram vocabulary size of 96k. In total, PEGASUS$_{\text{LARGE}}$ had 568M parameters.

To encourage the model to copy, which is an important aspect of the more extractive datasets, we left 20% of selected sentences unchanged in the input instead of replacing with [MASK1]. We increased the GSR to 45% to achieve a similar number of “gaps” as the optimal 30% found above. We reported the performance of the models pre-trained on HugeNews and C4 separately. We conducted a simple hyper-parameter sweep of learning rate and length penalty,
alpha, when fine-tuning PEGASUS\textsubscript{LARGE} on each downstream dataset.

CNN/DailyMail, Multi-News, arXiv, PubMed, BIGPATENT datasets contain input documents longer than the maximum input length ($L_{\text{input}} = 512$ tokens) in pre-training. This would present a problem for position embeddings which would never be updated for longer input lengths, but we confirm the postulation that sinusoidal positional encodings (Vaswani et al., 2017) generalize well when fine-tuning PEGASUS\textsubscript{LARGE} beyond the input lengths observed in training up to $L_{\text{input}} = 1024$ tokens. Since average input length in BIGPATENT, arXiv, PubMed and Multi-News are well beyond 1024 tokens, further scaling up $L_{\text{input}}$ or applying a two-stage approach (Liu et al., 2018) may improve performance even more, although this is outside the scope of this work.

Tables 1 and 2 show the performance improvements of PEGASUS\textsubscript{BASE} and PEGASUS\textsubscript{LARGE} on downstream datasets. While PEGASUS\textsubscript{BASE} exceeded current state-of-the-art on many datasets, PEGASUS\textsubscript{LARGE} achieved better than state-of-the-art results on all downstream datasets using PEGASUS\textsubscript{HugeNews}, although C4 performed better on WikiHow.

The improvement from a Transformer model without pre-training (Transformer\textsubscript{BASE}) to PEGASUS\textsubscript{LARGE} was more significant on smaller datasets. For example, the ROUGE2-F1 scores nearly tripled on AESLC and quintupled on Reddit TIFU. The large jumps in performance suggest that small text summarization datasets benefit the most from pre-training. We further investigate low resource summarization in Section 6.3.

### 6.3 Zero and Low-Resource Summarization

In real-world practice, it is often difficult to collect a large number of supervised examples to train or fine-tune a summarization model. To simulate the low-resource summarization setting, we picked the first $10^k$ ($k = 1, 2, 3, 4$) training examples from each dataset to fine-tune PEGASUS\textsubscript{LARGE} (HugeNews). We fine-tuned the models up to 2000 steps with batch size 256, learning rate 0.0005, and picked the checkpoint with best validation performance. In Figure 6, in 8 out of 12 datasets, with just 100 examples...
Table 3: Human evaluation side-by-side results on Likert (1-5) scale (higher is better). Scores are bolded if they are not worse than human-level performance by \( p < 0.01 \).

<table>
<thead>
<tr>
<th>Datasets</th>
<th>XSum</th>
<th>CNN/DailyMail</th>
<th>Reddit TIFU</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Experiment 1: pretrain comparison</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Human-written</td>
<td>3.0 (+)</td>
<td>3.1 (+)</td>
<td>3.2 (+)</td>
</tr>
<tr>
<td>PEGASUS_{HugeNews}</td>
<td>3.0 (0.04)</td>
<td>2.6 (0.0001)</td>
<td>3.2 (0.07)</td>
</tr>
<tr>
<td>PEGASUS_{C4}</td>
<td>3.1 (0.7)</td>
<td>3.2 (0.009)</td>
<td>3.4 (0.3)</td>
</tr>
<tr>
<td>Transformer_{BASE}</td>
<td>2.0 (3e-08)</td>
<td>2.9 (0.06)</td>
<td>1.4 (3e-23)</td>
</tr>
<tr>
<td><strong>Experiment 2: few resource</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Human-written</td>
<td>3.2 (+)</td>
<td>3.2 (+)</td>
<td>3.3 (+)</td>
</tr>
<tr>
<td>PEGASUS_{HugeNews} (10 examples)</td>
<td>2.8 (0.1)</td>
<td>2.4 (0.003)</td>
<td>2.6 (0.006)</td>
</tr>
<tr>
<td>PEGASUS_{HugeNews} (100 examples)</td>
<td>3.2 (0.5)</td>
<td>3.4 (0.08)</td>
<td>2.1 (4e-8)</td>
</tr>
<tr>
<td>PEGASUS_{HugeNews} (1000 examples)</td>
<td>3.4 (0.3)</td>
<td>3.6 (0.07)</td>
<td>2.7 (0.01)</td>
</tr>
<tr>
<td>PEGASUS_{HugeNews} (full supervision)</td>
<td>3.4 (0.3)</td>
<td>3.3 (0.1)</td>
<td>2.8 (0.05)</td>
</tr>
</tbody>
</table>

PEGASUS_{LARGE} could be fine-tuned to generate summaries at comparable quality to Transformer_{BASE} trained on the full supervised datasets ranging from 20k to 200k examples. PEGASUS_{LARGE} also beat previous state-of-the-art results on 6 out of 12 datasets with only 1000 fine-tuning examples.

On CNN/DailyMail, with half the number of parameters PEGASUS_{LARGE} demonstrated much better zero-shot (ROUGE2-F=13.28) performance than GPT-2 (ROUGE2-F=8.27). Using only 1000 examples, PEGASUS_{LARGE} achieved ROUGE2-F of 19.35, much higher than the 13.1 obtained in [Khandelwal et al., 2019] with 3000 examples.

6.4 Qualitative Observations and Human Evaluation

Overall, we observed high-linguistic quality (in terms of fluency and coherence), closely emulating the style of ground-truth summaries. While some previous work suggested that maximum likelihood training results in repetitive text in model outputs ([Welleck et al., 2019]) we found this to be rare in our outputs and did not require additional countermeasures to mitigate dis-fluencies.

Although ROUGE clearly has its drawbacks ([Kryscinski et al., 2019]), over-penalizing abstractive approaches compared to extractive ones and having no sense of linguistic quality, we found that choosing perplexity-optimized models using aggregated ROUGE (rather than directly optimizing ROUGE as in [Paulus et al., 2017]) resulted in qualitatively good models. Randomly sampled (by a program) model decodes across all datasets and a broad range of ROUGE scores can be found in Appendix [I]. We found that even low-ROUGE model summaries often were high-quality. Figure [G.1]

To assess how close PEGASUS_{LARGE} is to human performance we conducted human evaluation experiments on Amazon Mechanical Turk comparing model summaries with (human) reference summaries given the input document. The examples were drawn from the XSum, CNN/DailyMail, and Reddit TIFU datasets; the first two were chosen due to their popularity in past work, and the third was chosen for its significant difference in style. Workers were asked to rate the summaries on a 1-5 scale, with higher being better (full experiment details provided in Appendix [E] and a paired t-test was used to assess whether scores were significantly different from human.

In the first experiment, PEGASUS_{LARGE} (HugeNews), PEGASUS_{LARGE} (C4), and Transformer_{BASE} were compared with reference summaries; in the second experiment, PEGASUS_{LARGE} (HugeNews) fine-tuned using 10, 100, 1000, and all supervised examples were compared with references; the results are shown in Table 3. According to the significance level of \( p < 0.01 \), both PEGASUS_{LARGE} (HugeNews) and PEGASUS_{LARGE} (C4) outputs were at least as good as the reference summaries in all cases. Even at low-levels of supervision PEGASUS_{LARGE} (HugeNews) was not measurably worse than human summaries on XSum and CNN/DailyMail. In the Reddit TIFU case, however, perhaps due to its diverse writing styles, human performance required full supervision.
6.5 Test-set Overlap with Pre-training Corpus

The pre-training corpora are a large collection of documents from the Internet and potentially have overlap with the downstream test sets. In this section, we measured the extent of overlap between the pre-training corpus and downstream datasets. We also studied if the pre-trained model was able to exploit memorization to achieve higher performance on the downstream datasets.

To measure the overlap, we calculated similarities between all pairs of downstream test set targets and pre-training documents. We use the ROUGE-2 recall as a similarity measure (common 2-grams / test set targets 2-grams). It is not necessarily exact match even if the similarity score is 1.0. We filtered all test set examples that have similarity to any pre-training example above a threshold, and recalculated the ROUGE scores on the remaining test set. In Figure 7, we conducted this study on the pre-training corpus C4 and test set of XSum, CNN/Dailymail, Reddit TIFU and WikiHow, with a similarity threshold of 1.0 and 0.8. Results show that only XSum has significant amount of overlap 15% to 20%, and filtering those examples does not change ROUGE scores more than 1%. We also manually examined those overlapped examples with similarity of 1.0, and found that the models produce very different summaries compared to the human written ones, suggesting that there was no clear memorization.

80% 85% 90% 95% 100%
post-filter remaining data
-1% -0.5% 0% 0.5% 1%
average change in ROUGE-1/2/L

Figure 7: Percentage of overlap between C4 and downstream test sets, and ROUGE score changes after removing those overlapped examples in test sets.

6.6 Additional PEGASUS_{LARGE} Improvements

Following our experiments on PEGASUS_{LARGE} pre-trained on C4 and HugeNews, we pre-trained a PEGASUS_{LARGE} model on both corpora and stochastically sampled important sentences. The PEGASUS_{LARGE} (mixed, stochastic) model includes the changes: (1) The model was pre-trained on the mixture of C4 and HugeNews weighted by their number of examples. (2) The model dynamically chose gap sentences ratio uniformly between 15%-45%. (3) Importance sentences were stochastically sampled with 20% uniform noise on their scores. (4) The model was pre-trained for 1.5M steps instead of 500k steps, as we observed slower convergence of pre-training perplexity. (5) The SentencePiece tokenizer was updated to encode the newline character. The PEGASUS_{LARGE} (mixed, stochastic) model achieved best results on almost all downstream tasks, as shown in Table 4.

7 Conclusion

In this work, we proposed PEGASUS, a sequence-to-sequence model with gap-sentences generation as a pre-training objective tailored for abstractive text summarization. We studied several gap-sentence selection methods and identified principle sentence selection as the optimal strategy. We demonstrated the effects of the pre-training corpora, gap-sentences ratios, vocabulary sizes and scaled up the best configuration to achieve state-of-the-art results on all 12 diverse downstream datasets considered. We also showed that our model was able to adapt to unseen summarization datasets very quickly, achieving strong results in as little as 1000 examples. We finally showed our model summaries achieved human performance on multiple datasets using human evaluation.

8 Code and Model Checkpoints Release

The training code and instructions for using model checkpoints can be found at

https://github.com/google-research/pegasus

Acknowledgments

We thank Anastassia Kornilova, Eva Sharma, Shashi Narayan, Adam Roberts, Etienne Pot, and the Google News team for assistance with datasets, and Carey Radebaugh, David Grangier, Doug Eck, and Samy Bengio for reviewing the manuscript.
References


