Leveraging Pre-trained Checkpoints for Sequence Generation Tasks

Sascha Rothe
Google Research
rothe@google.com

Shashi Narayan
Google Research
shashinarayan@google.com

Aliaksei Severyn
Google Research
severyn@google.com

Abstract

Unsupervised pre-training of large neural models has recently revolutionized Natural Language Processing. By warm-starting from the publicly released checkpoints, NLP practitioners have pushed the state-of-the-art on multiple benchmarks while saving significant amounts of compute time. So far the focus has been mainly on the Natural Language Understanding tasks. In this paper, we demonstrate the efficacy of pre-trained checkpoints for Sequence Generation. We developed a Transformer-based sequence-to-sequence model that is compatible with publicly available pre-trained BERT, GPT-2, and RoBERTa checkpoints and conducted an extensive empirical study on the utility of initializing our model, both encoder and decoder, with these checkpoints. Our models result in new state-of-the-art results on Machine Translation, Text Summarization, Sentence Splitting, and Sentence Fusion.

1 Introduction

Unsupervised and self-supervised pre-training methods, such as ELMo (Peters et al., 2018), ULMFiT (Howard and Ruder, 2018), and more recently BERT (Devlin et al., 2019), GPT and GPT-2 (Radford et al., 2018, 2019), XLNet (Yang et al., 2019), and RoBERTa (Liu et al., 2019) have established a qualitatively new level of baseline performance for many widely used Natural Language Understanding (NLU) benchmarks including some of the most popular, like GLUE (Williams et al., 2018) and SQuAD (Rajpurkar et al., 2018).

The most appealing part about this massive shift towards using large architectures pre-trained on large collections of texts is that the pre-trained checkpoints along with the inference code are made freely available. This saves hundreds of TPU/GPU hours, as warm-starting a model from a pre-trained checkpoint typically requires orders of magnitude fewer fine-tuning steps while delivering significant performance boosts. More importantly, the ability to bootstrap from a state-of-the-art performing model such as BERT (Devlin et al., 2019) motivates the community to greatly speed up the progress towards developing better and easily reusable NLU systems.

While we continue to observe an increasing number of papers building on top of BERT and/or GPT models reporting encouraging improvements on GLUE, SQuAD, and other similar benchmarks, very little attention has been paid to using these pre-trained models to warm-start sequence-to-sequence (seq2seq) models. It has been argued that the pre-training objective used by BERT is not well-suited for tasks that require decoding texts, for example, conditional text generation in machine translation and summarization (Yang et al., 2019). Nevertheless, it remains unclear to what extent using such large models pre-trained on large collections of text can be beneficial to warm-start seq2seq generation models.

In this paper, we report on a Transformer-based seq2seq model that is compatible with publicly available pre-trained BERT, GPT-2, and RoBERTa checkpoints. We aim to provide an empirical answer to the following research question: *What is the best way to leverage publicly available pre-trained checkpoints for warm-starting sequence generation models?* For example, one could imagine using a BERT checkpoint to initialize the encoder for better input understanding and choosing GPT-2 model as the decoder for better text generation. One of the main contributions of this paper is that we rigorously experiment with a large number of different settings to combine BERT, GPT, and RoBERTa pre-trained checkpoints to initialize our Transformer-based model. We report results on three canonical conditional text generation tasks of increasing complexity: sentence-level fusion (DiscoFuse, Geva et al., 2019) and splitting (WikiSplit, Botha et al., 2018),

Transactions of the Association for Computational Linguistics, vol. 8, pp. 264–280, 2020. https://doi.org/10.1162/tacl_a_00313
Action Editor: Wenjie (Maggie) Li. Submission batch: 9/2019; Revision batch: 12/2019; Published 6/2020.

WMT14 En↔De machine translation using most common eval sets: newstest2014 and newstest2016, and abstractive summarization using three datasets: Gigaword (Napoles et al., 2012), CNN and DailyMail (Hermann et al., 2015), and BBC extreme (Narayan et al., 2018a).

Our models report significant improvements over randomly initialized models, demonstrating the benefit of leveraging unsupervised pre-trained models. More importantly, this simple strategy results in new state-of-the-art results on machine translation, text summarization, sentence splitting, and sentence fusion. Our results also demonstrate that a pre-trained encoder is an essential component for sequence generation tasks and often these tasks benefit from sharing the weights between the encoder and the decoder. Overall, we have run over 300 experiments spending thousands of TPU v3 hours to better accommodate the language modeling and understanding capabilities of these pre-trained models for text generation. We believe that NLP researchers and practitioners will derive actionable insights from our findings when tackling various seq2seq tasks.

The code to query our models and predictions on various benchmarks will be available at `https://github.com/google-research/google-research/tree/master/bertseq2seq`.

2 Models and Pre-trained Checkpoints

BERT was primarily developed for encoding text representations for NLU tasks (encoder-only architecture), whereas GPT-2 (Radford et al., 2019), was primarily developed as a decoder-only architecture for language modeling. Our model uses a seq2seq architecture with encoder and decoder both composed of Transformer layers (Vaswani et al., 2017). For the encoder, we inherit the BERT Transformer layer implementations (Devlin et al., 2019), which differs slightly from the canonical Transformer layer (Vaswani et al., 2017); BERT uses a GELU activation (Hendrycks and Gimpel, 2016) rather than the standard RELU. If not stated otherwise, the implementation of the decoder layers are also identical to the BERT implementation with two adjustments. First, the self-attention mechanism is masked to look only at the left context. Secondly, we add an encoder-decoder attention mechanism. Note, that if the model was randomly initialized, we found no difference between a BERT compatible decoder and a GPT-2 compatible decoder.

Most of the models use the base checkpoint and therefore have 12 layers, a hidden size of 768, filter size of 3,072, and 12 attention heads. We chose the best-performing model and also collect numbers using larger pre-trained checkpoints. These models have 24 layers, a hidden size of 1,024, filter size of 4,096, and 16 attention heads.

All models were fine-tuned on the target task using Adam with a learning rate of 0.05. We used a linear learning rate warmup with 40k steps, normalization by the square root of the hidden size, and a square root decay. We did not perform any tuning of these hyperparameters (except for §5). The batch size and the number of training steps will be reported for each task individually.

BERT Checkpoints. We tokenize our text using the WordPiece (Wu et al., 2016) to match the BERT pre-trained vocabulary. Depending on the experiment, we use one of the following publicly available checkpoints: BERT-Base Cased, BERT-Base Uncased, BERT-Base Multilingual Cased (Devlin et al., 2019).[1] The first two checkpoints have a vocabulary size of around ∼30k word-pieces, whereas the multilingual checkpoint has a much larger vocabulary size of ∼110k. BERT also trains positional embeddings for up to 512 positions, which is the maximum input and output length in all experiments.

GPT-2 Checkpoints. We tokenize our text using the SentencePieces (Kudo and Richardson, 2018) to match the GPT-2 pre-trained vocabulary.[2] Note that, although the available checkpoint is frequently called 117M, which suggests the same number of parameters, we count 125M parameters in the checkpoint. This is the smallest architecture they trained, and the number of layers, hidden size, and filter size are comparable to BERT-Base. The model was trained mainly on English data but does contain some foreign language. The vocabulary size is ∼50k. While GPT-2 has positional embeddings for up to 1,024 positions, we only use the first 512 to make the results comparable with BERT.

RoBERTa Checkpoints. RoBERTa (Liu et al., 2019) is trained using PyTorch, but we found that the learned parameters are fully compatible

[1] BERT checkpoints are available at `https://github.com/google-research/bert`.

[2] GPT-2 checkpoints are available at `https://github.com/openai/gpt-2`.

	total	embed.	init.	random
RND2RND	221M	23M	0	221M
BERT2RND	221M	23M	109M	112M
RND2BERT	221M	23M	109M	26M
BERT2BERT	221M	23M	195M	26M
BERTSHARE	136M	23M	109M	26M
ROBERTASHARE	152M	39M	125M	26M
GPT	125M	39M	125M	0
RND2GPT	238M	39M	125M	114M
BERT2GPT	260M	62M	234M	26M
ROBERTA2GPT	276M	78M	250M	26M

Table 1: The number of total trainable parameters, embedding parameters, and parameters initialized from the checkpoint vs. randomly. The BERT/GPT-2 embeddings have 23M/39M parameters. The encoder-decoder attention accounts for 26M parameters.

with the existing TensorFlow BERT architectures with some minor adjustments.[3] The vocabulary treatment in RoBERTa is compatible with the SentencePiece tokenization in GPT-2.[4] As the conceptual differences between BERT and RoBERTa are minor, we might use BERT as a hypernym to address both pretraining methods in this paper.

3 Investigated Model Variants

In this section, we describe several combinations of model initialization. The number of total trainable parameters, the number of embedding parameters, and the number of parameters initialized from the checkpoint vs. randomly are shown in Table 1.

RND2RND A Transformer encoder-decoder architecture with all weights initialized randomly.

BERT2RND A BERT-initialized encoder paired with a randomly initialized decoder. Encoder and decoder share the embedding matrix initialized from a checkpoint.

RND2BERT A randomly initialized encoder paired with a BERT-initialized decoder. To perform autoregressive decoding, we mask the bidirectional self-attention mechanism of BERT to look only at the left context.

BERT2BERT A BERT-initialized encoder paired with a BERT-initialized decoder. All weights are initialized from a public BERT checkpoint. The only variable that is initialized randomly is the encoder-decoder attention.

BERTSHARE Like BERT2BERT, but the parameters between encoder and decoder are shared. This greatly reduces the memory footprint of the model (136M vs. 221M parameters). Additionally, we experimented with a layer-wise attention mechanism (He et al., 2018), but obtained nearly identical numbers on most tasks.

ROBERTASHARE Same as BERTSHARE, but the shared encoder and decoder are initialized with the public RoBERTa checkpoint.

GPT A decoder-only architecture. We treat the input as a conditioning prefix of a language model. The decoder is warm-started with a public GPT-2 checkpoint. Similarly to BERTSHARE and ROBERTASHARE, the memory footprint of this model is smaller compared to an encoder-decoder setup (125M parameters).

RND2GPT A randomly initialized encoder paired with a GPT-2-compatible decoder. We warm-start the decoder and the embedding matrix with a public GPT-2 checkpoint.

BERT2GPT A BERT-compatible encoder paired with a GPT-2-compatible decoder. We warm-start both sides with the two separate, BERT and GPT-2, public checkpoints. We use the BERT vocabulary for the input and the GPT-2 vocabulary for the output.

ROBERTA2GPT Same as BERT2GPT, but we use a public RoBERTa checkpoint to warm-start the encoder. RoBERTa was trained using the GPT-2 vocabulary so we can use it for input and output. Note that although the vocabulary is shared, this model still has two embeddings matrices, one for the input and one for the output.

The pre-training objective in the BERT models learns to predict a masked token using the bidirectional representation of the input text (Devlin et al., 2019; Liu et al., 2019). Our decoder, even when initialized with the BERT or RoBERTa checkpoints, always generates the output text in an autoregressive fashion as in Tranformers (Vaswani et al., 2017) and GPT-2 (Radford et al., 2019).

[3] More specifically: a) the variable names have to be adjusted; b) the weight and bias variables of the attention mechanism have to be splitted into query, key, and values; c) all variables except the embedding matrices have to be transposed.

[4] RoBERTa checkpoints are available at `https://github.com/pytorch/fairseq`.

DiscoFuse	100%		10%	1%
	Exact	SARI	SARI	SARI
(Geva et al., 2019)	51.1	84.5	–	–
Initialized with the base checkpoint (12 layers)				
ROBERTA2GPT	**65.6**	**89.9**	**87.1**	80.3
ROBERTASHARE	65.3	89.7	86.9	**81.2**
BERT2BERT	63.9	89.3	86.1	**81.2**
BERT2RND	63.9	89.3	86.1	80.3
BERTSHARE	63.9	89.2	86.0	80.8
BERT2GPT	61.5	88.4	84.1	70.2
GPT	60.4	88.0	82.9	74.5
RND2BERT	60.0	87.6	82.1	72.8
RND2RND	58.3	86.9	81.5	69.3
RND2GPT	57.6	86.5	81.4	70.6
Initialized with the large checkpoint (24 layers)				
ROBERTASHARE	**66.6**	**90.3**	**87.7**	**81.5**
BERTSHARE	65.3	89.9	86.6	81.4

Table 2: Results of different models and initialization techniques on DiscoFuse and subsampled training sets. Blockwise sorted by SARI score on 100% of the training set.

WikiSplit	Exact	SARI	BLEU
(Botha et al., 2018)	14.3	61.5	76.4
Initialized with the base checkpoint (12 layers)			
BERTSHARE	**16.3**	**63.5**	**77.2**
ROBERTASHARE	16.1	63.4	77.1
BERT2BERT	15.6	63.2	77.0
ROBERTA2GPT	15.1	63.2	76.8
BERT2RND	15.9	63.1	76.9
BERT2GPT	14.6	62.4	76.5
RND2BERT	15.2	61.8	76.5
RND2RND	14.6	61.7	76.3
RND2GPT	14.2	61.3	76.2
GPT	14.2	61.1	75.8
Initialized with the large checkpoint (24 layers)			
ROBERTASHARE	16.4	**63.8**	**77.4**
BERTSHARE	**16.6**	63.7	77.3

Table 3: Results of different models and initialization setups on WikiSplit. Blockwise sorted by SARI score.

We performed the bulk of our experiments on the 12-layer checkpoints of BERT, GPT-2, and RoBERTa, assuming that the findings will also hold for the 24-layer checkpoints. We chose BERTSHARE, ROBERTASHARE and ROBERTA to also report numbers using the 24-layer public pretrained checkpoints. We also experimented with the GPT setup with 24 layers and 345M parameters but as we did not achieve any better results we excluded this from the paper.

4 Experiments and Results

4.1 Sentence Fusion

Sentence Fusion is the problem of combining multiple sentences into a single coherent sentence. We use the ''balanced Wikipedia'' portion of the DiscoFuse dataset (Geva et al., 2019) for our experiments with 4.5M fusion examples in the training set. The evaluation set has 50k examples. Because of the size of this evaluation set, even small changes are statistically significant. For this reason, we have solely chosen this dataset for additional experiments described at the end of the paper.

Training was done for 300k steps with a global batch size of 256. The input and output are padded to a length of 128, which covers 100% of the training, evaluation, and test data. We report SARI (Xu et al., 2016)[5] and the exact match accuracy. The results can be seen in Table 2. Previous state-of-the-art results by Geva et al. (2019) used the vanilla transformer model by Vaswani et al. (2017), with only 7 layers. All models with initialized encoders outperform the baseline by a large margin, with a SARI score of 89.3 compared with 86.9 (BERT2RND vs. RND2RND). To measure the effect on smaller training sets, we randomly subsample the training data down to 10% and 1%, (i.e., 450k and 45k training examples, respectively). First, we notice, that performance comparable to the baseline is achieved even when training on only 10% of the training data (ROBERTASHARE vs. ROBERTASHARE). Secondly, when using only 1% of the training data, setups with fewer randomly initialized parameters (BERT2BERT vs. BERT2RND) perform better. The best performing 12-layer setup is ROBERTA2GPT with a SARI score of 89.9 only outperformed by 24-layer setup of ROBERTASHARE with a SARI score of 90.3.

4.2 Split and Rephrase

The reverse task of sentence fusion is the split-and-rephrase task, which requires rewriting a long

[5]SARI is a lexical similarity metric that compares the model's output to multiple references and the input in order to assess the model's ability to add, delete, and keep an n-gram. Its implementation is available at: https://github.com/tensorflow/tensor2tensor/blob/master/tensor2tensor/utils/sari_hook.py.

sentence into two or more coherent short sentences (Narayan et al., 2017). We use the WikiSplit dataset (Botha et al., 2018), which consists of 1M examples of sentence splits extracted from the Wikipedia edit history, and follow the training/test split suggested by the authors. Training was done for 300k steps with a global batch size of 256. The input and output are padded to a length of 128, which covers 100% of the training, evaluation, and test data. As in Botha et al. (2018), we report corpus-level BLEU,[6] the exact match accuracy, and SARI score. Previous state-of-the-art results by Botha et al. (2018) used a bi-directional LSTM with a copy mechanism (Aharoni and Goldberg, 2018). Analogous to the DiscoFuse task we observe that initializing the encoder improves the model the most (Table 3). The shared encoder-decoder setup of BERTSHARE outperforms all other setups. For the larger models with 24 layers, we observed a small over-fitting after 100k steps (˜25 epochs), and therefore stop the training early. BERTSHARE and ROBERTASHARE perform on par and both outperform their 12-layer counterpart.

4.3 Machine Translation

We test our setups on the most common benchmark in machine translation—WMT 2014 English ↔ German task—using newstest2014 and newstest2016 eval sets. We use the same hyper-parameter settings as in the previous experiments. We limit the input and output lengths to 128 tokens each. We used a global batch size of 256 and train for 30 epochs. Decoding was done with beam size of 4 and the default value for the sentence length penalty set to $\alpha = 0.6$. We report uncased BLEU-4 scores.[7]

In Table 4, we first report the baseline scores for the original Transformer model Vaswani et al. (2017) and our Transformer implementation[8] with the same hyper parameters. In both cases, we use the encoder and decoder with 6 layers and the 32k wordpiece vocabulary extracted from the WMT14 training set. Our implementation obtains slightly higher scores than the original implementation.

The middle section of Table 4 reports the results for various initialization schema using BERT and GPT-2 pre-trained checkpoints. Note that here all models have encoders and decoders with 12 layers. For BERT models, we use the BERT-Base Multilingual Cased checkpoint to initialize the encoder or the decoder or both, as the task involves one non-English language. This checkpoint has been pre-trained on 108 languages using a multilingual Wikipedia dump with a vocabulary of 110k wordpieces. First, we observe that initializing the model with the BERT checkpoint is most beneficial on the encoder side; our observation is in line with Yang et al. (2019). Furthermore, models initialized with the BERT checkpoint receive a significant boost: BERT2RND compared to the no-initialization RND2RND setup scores higher by +4 points on En→De and +3.6 points on De→En on newstest2014. Contrary to the WikiSplit and DiscoFuse task, sharing the encoder and decoder variables did not give an additional boost. This is most likely because a) model capacity is an important factor in MT and b) encoder and decoder have to deal with different grammar and vocabulary.

GPT-based models (RND2GPT, GPT, and BERT2GPT) do not perform nearly as well, especially when GPT is used as the decoder and the target language is German. This is because the GPT model comes with an English vocabulary and has been pre-trained mainly on English text. Hence, we report the scores for GPT in the En→De setting in gray.

Customized BERT Checkpoint. For this experiment we did not include RoBERTa, as the public checkpoint is available for English only. Instead, we train our own checkpoint. We also observe that our implementation of the baseline Transformer, as well as RND2RND setup, which uses no initialization, more weakly weaker on newstest2014 compared with the Transformer baselines (with 6 layers and the 32k wordpiece vocabulary) we report in the top section of Table 4. We conjecture that the differences might be due to the larger 110k wordpiece vocabulary trained to handle 104 languages from Wikipedia dump,

[6] We use NLTK v3.2.2 with case-sensitive scoring to estimate BLEU scores.

[7] We use a script from the Tensorflow Official Transformer implementation `https://github.com/tensorflow/models/tree/master/nlp/transformer`. Note that, differently from the `https://github.com/tensorflow/tensor2tensor/blob/master/tensor2tensor/utils/get_ende_bleu.sh` used by Vaswani et al. (2017), this script does not split noun compounds, but we normalize utf-8 quotes to ascii quotes as we noted that our pre-processed training set contains only ascii quotes.

[8] We use Transformer layers from the official BERT implementation which have small differences from Vaswani et al. (2017).

	newstest2014		newstest2016	
	En→De	De→En	En→De	De→En
(Vaswani et al., 2017)	27.3	–	–	–
Transformer (ours)	28.1	31.4	33.5	37.9
KERMIT(Chan et al., 2019)	28.7	31.4	–	–
(Shaw et al., 2018)	29.2	–	–	–
(Edunov et al., 2018)*	**35.0** (33.8)	–	–	–
Initialized with public checkpoints (12 layers) and vocabulary				
Transformer (ours)	23.7	26.6	31.6	35.8
RND2RND	26.0	29.1	32.4	36.7
BERT2RND	**30.1**	**32.7**	34.4	**39.6**
RND2BERT	27.2	30.4	33.2	37.5
BERT2BERT	**30.1**	**32.7**	**34.6**	39.3
BERTSHARE	29.6	32.6	34.4	**39.6**
GPT	16.4	21.5	22.4	27.7
RND2GPT	19.6	23.2	24.2	28.5
BERT2GPT	23.2	31.4	28.1	37.0
Initialized with a custom BERT checkpoint (12 layers) and vocabulary				
BERT2RND	**30.6**	33.5	35.1	**40.2**
BERTSHARE	30.5	**33.6**	**35.5**	40.1
Initialized with a custom BERT checkpoint (24 layers) and vocabulary				
BERT2RND	**31.7**	**34.2**	**35.6**	**41.1**
BERTSHARE	30.5	33.8	35.4	40.9

Table 4: Uncased BLEU-4 scores on WMT14 English ↔ German newstest2014 and newstest2016 test sets. Models in the middle section use the 110k wordpiece vocabulary that comes with the multilingual BERT checkpoint. In the bottom section, we use the native 32k wordpiece vocabulary extracted from WMT14 train set and a BERT checkpoint pre-trained only on English and German subset of Wikipedia. * Leveraging a large number of additional parallel sentence pairs obtained with back-translation; we include this score as a reference to the highest achieved result on newstest2014. The GPT-2 results for En→De (where the GPT-2 initialized decoder is used to decode targets in De) are grayed out as they are a priori penalizing for GPT-2, which was only pretrained on En texts.

which is suboptimal for WMT14 data and leads to inferior results. To verify this conjecture, we perform the following experiment: We use the 32k wordpiece vocabulary extracted from the WMT14 En ↔ De training set (same as used in the top section of Table 4) and pre-train a BERT model on the English and German subset of the Wikipedia dump in the same way as the multilingual BERT checkpoint was obtained. We initialize our best-performing setups, BERT2RND and BERTSHARE, with this checkpoint (the third block of Table 4). This provides a further +0.5 (En ↔ De) and +0.8 (De ↔ En) BLEU improvements on newstest2014, and, +1.1 and +0.7 on newstest2016, yielding an overall very strong performance on the challenging WMT14 task. Experiments with the larger models (the last block) show further improvements of up to +1.1 BLEU points.

Edunov et al. (2018) report better results when they augment the training set with a massive amount of back-translated sentence pairs. To the best of our knowledge, among the approaches that only leverage parallel data from WMT14, our results are state-of-the-art on both newstest2014 and newstest2016.

4.4 Abstractive Summarization

Document summarization is the task of producing a short version of a document while preserving its salient information content. We evaluate our setups on three different summarization datasets of varying characteristics: Gigaword (Napoles et al., 2012), CNN and DailyMail (Hermann et al., 2015), and BBC extreme (Narayan et al., 2018a). The Gigaword dataset focuses on abstractive sentence summarization with a total of 3.8M sentence-summary training pairs. The other two datasets focus on single-document summarization: The CNN/DailyMail dataset consists of 287k document–summary pairs, whereas the BBC dataset consists of 204k document-summary pairs. The CNN/DailyMail summaries are in the form of bullet-point story highlights and exhibit a high degree of extraction, requiring the models to learn to copy from the source documents. The BBC summaries, on the other hand, are extreme in that the documents are summarized into single-sentence summaries. These summaries demonstrate a high level of abstractiveness, and generating them automatically requires document-level inference, abstraction, and paraphrasing.

In all three cases, we did not anonymize entities. We worked on the original cased versions of the CNN/DailyMail and BBC datasets. For Gigaword we used the lowercased version to match the requirements of the publicly available lowercased test set. During training, the input documents were truncated to 512 tokens for the CNN/DailyMail and BBC, and to 128 tokens for Gigaword. Similarly, the length of the summaries was limited to 128 tokens for CNN/DailyMail, 64 for BBC, and 32 for Gigaword. We used a global batch size of 128 document–summary pairs for CNN/DailyMail and BBC, and 256 document–summary pairs for Gigaword. We adapted to different number of training steps depending on the training data sizes. Models were trained for 500k, 300k, and 200k steps for the Gigaword, CNN/DailyMail, and BBC summarization datasets respectively. In all cases, we used the standard publicly available test sets; these consists of 1951 instances for Gigaword, 11,490 for CNN/DailyMail, and 11,334 for BBC. We report on the ROUGE F_1 scores (Lin and Hovy, 2003); in particular, we report on ROUGE-1 and ROUGE-2 for informativeness and ROUGE-L for fluency in Table 5.

Document Understanding. All BERT encoder-based setups (i.e., BERT2RND, BERTSHARE, ROBERTASHARE, and BERT2BERT) outperform the baseline RND2RND by a large margin. The improvements of the RND2BERT setup, where only the decoder is initialized, are narrow. These results overall validate the significance of document representation in the encoder-decoder framework for summarization. On the BBC extreme summarization in particular, these four models achieve on average +6.85 point improvement in ROUGE-L compared with the RND2RND setup. Our results demonstrate that the models with better document representations are better in generating extreme summaries that require document-level inference and abstraction. For the extractive highlights in the CNN/DailyMail dataset, these models show an improvement of +3.53 ROUGE-L points over the RND2RND baseline. For Gigaword, where the input is a single sentence, the improvements are minimal (average of +1.02 ROUGE-L points). The BERTSHARE setup with shared encoder and decoder parameters achieves better performance than BERT2BERT on all three datasets. The gains are larger on the BBC dataset than on the Gigaword and CNN/DailyMail datasets. This is probably because the BBC summary sentences follow a distribution that is similar to that of the sentences in the document, whereas this is not necessarily the case for the Gigaword headlines and the CNN/DailyMail bullet-point highlights. ROBERTASHARE performs superior to BERTSHARE on the CNN/DailyMail and BBC datasets. ROBERTASHARE performs competitively to BERTSHARE on the Gigaword dataset where the task is to summarize sentences.

Summarization with GPT Checkpoints. GPT (decoder-only) performs better than RND2GPT, BERT2GPT or ROBERTA2GPT (encoder-decoder models) by a large margin for generating CNN/DailyMail extracts, but poorer for generating BBC abstracts. The encoder–decoder architecture where the input document is modeled separately is better equipped for document-level abstraction than the decoder-only architectures where the input document is a conditioning prefix of a language model. Initialization with different checkpoints (e.g., encoder with BERT and decoder with GPT in BERT2GPT) is not effective for document summarization; BERT2GPT and ROBERTA2GPT are inferior to RND2GPT on the

	Gigaword			CNN/DailyMail			BBC XSum		
	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
Lead	–	–	–	39.60	17.70	36.20	16.30	1.61	11.95
PtGen	–	–	–	39.53	17.28	36.38	29.70	9.21	23.24
ConvS2S	35.88	17.48	33.29	–	–	–	31.89	11.54	25.75
MMN	–	–	–	–	–	–	32.00	12.10	26.00
Bottom-Up	–	–	–	41.22	18.68	38.34	–	–	–
MASS	*38.73*	19.71	*35.96*	–	–	–	–	–	–
TransLM	–	–	–	39.65	17.74	36.85	–	–	–
UniLM	–	–	–	*43.47*	*20.30*	*40.63*	–	–	–
Initialized with the base checkpoint (12 layers)									
RND2RND	36.94	18.71	34.45	35.77	14.00	32.96	30.90	10.23	24.24
BERT2RND	37.71	19.26	35.26	38.74	17.76	35.95	38.42	15.83	30.80
RND2BERT	37.01	18.91	34.51	36.65	15.55	33.97	32.44	11.52	25.65
BERT2BERT	38.01	19.68	35.58	39.02	17.84	36.29	37.53	15.24	30.05
BERTSHARE	38.13	***19.81***	**35.62**	39.09	18.10	36.33	38.52	16.12	31.13
ROBERTASHARE	**38.21**	19.70	35.44	**40.10**	**18.95**	**37.39**	**39.87**	**17.50**	**32.37**
GPT	36.04	18.44	33.67	37.26	15.83	34.47	22.21	4.89	16.69
RND2GPT	36.21	18.39	33.83	32.08	8.81	29.03	28.48	8.77	22.30
BERT2GPT	36.77	18.23	34.24	25.20	4.96	22.99	27.79	8.37	21.91
ROBERTA2GPT	37.94	19.21	35.42	36.35	14.72	33.79	19.91	5.20	15.88
Initialized with the large checkpoint (24 layers)									
BERTSHARE	38.35	19.80	35.66	39.83	17.69	37.01	38.93	16.35	31.52
ROBERTASHARE	**38.62**	19.78	**35.94**	**40.31**	18.91	**37.62**	***41.45***	***18.79***	***33.90***

Table 5: Summarization results of different models and their initialization setups. We compare our setups (the bottom block) against both non-pre-trained (the top block) and pre-trained (the middle block) models on various datasets: the Lead baseline, PtGen (See et al., 2017), ConvS2S (Gehring et al., 2017), MMN (Kim et al., 2019), Bottom-Up (Gehrmann et al., 2018), MASS (Song et al., 2019), TransLM (Khandelwal et al., 2019), and UniLM (Dong et al., 2019). The Lead results for the CNN/DailyMail dataset is taken from Narayan et al. (2018b), whereas Lead, PtGen, and ConvS2S results on the BBC dataset are taken from Narayan et al. (2018a). Our best results are **boldfaced** and the best results on the datasets are *italicized*.

BBC dataset and BERT2GPT to RND2GPT on the CNN/DailyMail dataset. However, this is not the case with the Gigaword dataset, which has 3.8M training instances; BERT2GPT and ROBERTA2GPT perform better than RND2GPT.

ROBERTASHARE performs the best and is on par with the current state-of-the-art MASS model (Song et al., 2019) on the Gigaword dataset. The MASS model has an advantage of pre-training encoder-decoder attention from scratch, our proposed models use the publicly available pre-trained checkpoints and only fine-tune on the target task. It is not obvious how the masked seq2seq pre-training objective for sentence generation in the MASS model will be beneficial for tasks like document summarization. Our proposed models provide a generic alternative and can be easily adapted to various text generation tasks. The ROBERTASHARE setup sets a new state-of-the-art, outperforming all existing baselines by a large margin on the BBC extreme summarization task. The best model on the CNN/DailyMail dataset outperforms the Pointer Generator network (See et al., 2017) and the pre-trained single-decoder model with TransformerLM (Khandelwal et al., 2019). Our model, however, lags behind the Bottom-Up system (Gehrmann et al., 2018) with a task-specific module for content selection along with the copy

mechanism (Gu et al., 2016) and the UniLM model (Dong et al., 2019) with BERT-Large pre-trained for bidirectional, unidirectional and seq2seq language modeling objectives. The UniLM model is also fine-tuned with an additional extractive summarization objective to predict relevant sentences in the document; this objective could be beneficial to generate the CNN/DailyMail extracts.

5 Discussion on Ablation Studies

Combining Different Checkpoints. Combining BERT and GPT-2 into a single model (BERT2GPT) did not work and often underperformed than a randomly initialized baseline. This is presumably because the model has to learn two different vocabularies. This argument is backed by the fact that for MT, de→en the BERT2GPT setup performed well. For this task the vocabulary setting is in favor of this particular task, meaning that two vocabularies have to be learned anyways and the output is English, on which GPT-2 was trained. Because RoBERTa and GPT-2 share the same vocabulary, combining both into a single model (ROBERTA2GPT) showed strong results on several tasks but did not outperform a setup where RoBERTa is used in the encoder and decoder.

Tuning GPT-2 Based Models. We were surprised that setups using the GPT-2 checkpoint performed relatively poorly given that it is trained as a language model on a large corpus; our intuition was that GPT-2 initialized decoders will be strong natural language generators. To ensure that this was not due to an unfortunate choice of hyperparameters, we tuned the learning rate, the warmup steps, and the optimizer $\in$ {Adam, Adafactor} for the GPT-2 based setups (RND2GPT, GPT, BERT2GPT) on the DiscoFuse dataset. Naturally, this gave us slightly higher numbers but not at a magnitude that would suggest a previously suboptimal setting. Specifically, we obtained a SARI score of 88.8 compared with 88.4 for BERT2GPT, 88.1 compared with 88.0 for GPT, and 87.7 compared with 86.5 for RND2GPT.

Initializing Only Embeddings. We want to investigate the impact of the non-contextualized BERT and GPT-2 embeddings. This means we are initializing the transformer model with only the embedding matrices. The advantage of this setup would be that we could freely choose the model architecture and size and adapt it to a specific task. We found almost no improvement over the fully randomly initialized model RND2RND. Concretely, we compute a SARI score of 87.1 using the BERT embeddings and 87.0 using the GPT-2 embeddings, compared with 86.9 of the RND2RND baseline. We observe slightly higher improvements of up to 2 percentage points when training on only 10% of the training data.

Initializing Only Layers. Contrary to the previous paragraph, we want to investigate the effect of initializing everything but the word embedding matrix. The embedding matrix accounts for only 10% to 31% of all learnable parameters, and sometimes the vocabulary given from a public checkpoint might not be optimal for a certain task. In these cases, it would be nice to redefine the vocabulary while still leveraging the checkpoint. First, we remove the embeddings matrices from the warm-started variables and observe a drop of 1.7 points using the BERTSHARE setup and 11 points using the GPT setup (Table 6). The latter is probably due to the large vocab of the GPT-2 model, which now remains random-initialized. We then train a new BPE model with 16k tokens using the DiscoFuse training data (Kudo and Richardson, 2018; Sennrich et al., 2016). We observe almost no change on BERTSHARE, suggesting that the BERT vocabulary was already optimal for DiscoFuse. GPT, however, showed a significant improvement using this much smaller vocabulary but is still behind the fully initialized setup. Finally, we experimented with a more sensitive way of training the model, meaning that we fix all warm-started variables for 100k steps. During this pre-training phase, we only train the new word embeddings. After the pre-training, we fine-tune the entire model for another 300k steps. This training scheme resulted in an improvement of 0.5 for the BERTSHARE setup, but overall the number is still considerably behind the fully initialized setup. For GPT, this training scheme did not result in a satisfying training curve.

Initializing a Subset of Layers. Motivated by the results of using 24 layers, we want to investigate whether only a subset of these 24 layers can be used. To account for the larger hidden layer size (1,024 vs. 768) and filter size (4,096 vs. 3,072), we limit ourselves to using only 10 layers and the embedding matrix of this model. This model still

	BERTSHARE	GPT
DiscoFuse	89.3	88.0
- embeddings from checkpoint	87.5	77.0
+ task specific SentencePieces	87.5	84.2
+ pre-training SentencePieces	88.0	69.7

Table 6: SARI scores on the DiscoFuse dataset when experimenting with different embedding setups. Each row also includes the setups of all previous rows.

	Length	Repetitions	Quality
RND2RND	20.90	29.76	−0.103
RND2GPT	21.49	**16.28**	−0.303
BERTSHARE	20.71	27.03	−0.097
ROBERTASHARE	**21.70**	28.68	**0.153**
GOLD	24.61	4.66	0.347

Table 7: Qualitative and human evaluations of BBC extreme summaries. The lowest numbers for repetitions and the highest numbers for quality are *boldfaced*. See the text for details.

has more parameters then the base model (324M vs. 221M for BERT2BERT, 198M vs. 136M for BERTSHARE) but can be trained with the same batch size, in a comparable amount of time (3 min/1,000 iterations). As an initial experiment, we used the first 10 layers out of the large BERT checkpoint to initialize the BERTSHARE setup. This gave us a SARI score of 88.2 on DiscoFuse, compared with 89.3 when using the base checkpoint and compared with 87.0 when using the embeddings only (see Initializing Only Embeddings section). We then performed a hyperparameter search on the evaluation set using CMA-ES (Hansen, 2016) to find an optimal subset of layers to use. The best setup used the following layers: 9, 10, 13–18, 23, 24; and achieved a SARI score of 89.1. Although this is a remarkable improvement over using the first 10 layers, this setup is still outperformed by the base BERT model.

6 Analysis of Abstractive Summaries

Finally, we present a qualitative analysis of these models for text generation. In particular, we focus on extreme summarization, which assesses models ability to do document-level inference and abstraction. We evaluated summaries from arandomly initialized model (RND2RND) and from best performing models initialized with GPT checkpoints (RND2GPT), BERT checkpoints (BERTSHARE), and RoBERTa checkpoints (ROBERTASHARE). We also included GOLD summaries in our evaluation. Results are presented in Table 7.

Human Assessment of Summary Quality. The study was conducted on the Amazon Mechanical Turk platform using Best-Worst Scaling, a less labor-intensive alternative to paired comparisons (Louviere and Woodworth, 1991; Louviere et al., 2015). Our participants were presented with a document and summaries generated from two out of five systems (four models and gold summaries) and were asked to decide which summary was better than the other in order of informativeness (*does the summary capture important information in the document correctly and concisely?*) and fluency (*is the summary written in well-formed English?*) We randomly selected 40 documents from the XSum test set. We collected judgments from three different participants for each comparison. The order of summaries was randomized per document and the order of documents was randomized per participant. The score of a system was computed as the percentage of times it was chosen as best minus the percentage of times it was selected as worst. The scores range from −1 (worst) to 1 (best). See Figure 1 for a few sample predictions that were used in our human evaluation.

Our participants found the ROBERTASHARE summaries to be the best in terms of their overall quality; the BERTSHARE summaries ranked second after ROBERTASHARE. We further carried out pairwise comparisons between all models to assess whether system differences are statistically significant.[9] We did not observe significant differences between RND2RND and RND2GPT, RND2RND and BERTSHARE, and, ROBERTASHARE and GOLD. All other differences were statistically significant.

Summary Lengths and Repetitions. All models generated summaries of comparable lengths; the average length of summaries is 20.90 for RND2RND, 21.49 for RND2GPT, 20.71 for BERTSHARE, and 21.70 for ROBERTASHARE. ROBERTASHARE-produced summaries were closest to the GOLD summaries in terms of length (21.70 vs. 24.61).

[9]One-way ANOVA with post hoc Tukey HSD tests; $p < 0.01$.

RND2RND	The Queen has celebrated her 90th birthday with a message on social media about her 90th birthday.
RND2GPT	The Queen has celebrated her 90th birthday with a birthday celebration in Buckingham Palace.
BERTSHARE	The Queen has paid tribute to the Queen by sending a tweet saying she was "unwittingly unwittingly unwittingly.
ROBERTASHARE	The Queen has sent a twitter message for her 90th birthday on twitter.
GOLD	The Queen has tweeted her thanks to people who sent her 90th birthday messages on social media.
RND2RND	Sir Bradley Wiggins says he is "proud" of being involved in the use of a banned steroid against Sir Bradley Wiggins.
RND2GPT	Team Sky boss Sir Dave Brailsford says he is "disappointed" after team Sky agreed to change their contracts with team Sky.
BERTSHARE	Team Sky boss Sir Dave Brailsford says he is "proud" of his team's handling of doping in cycling.
ROBERTASHARE	Team Sky boss Dave Brailsford says he is "not proud" of his team's handling of allegations of wrong-doing in the sport.
GOLD	Team Sky boss Sir Dave Brailsford has said that his handling of the media following allegations against his team has made things a "damn sight worse".
RND2RND	A 19-year-old American singer has been shot dead by police in San Francisco.
RND2GPT	Police are investigating a shooting in the grounds of a music venue in Los Angeles.
BERTSHARE	US singer Chris Brown has been shot and wounded at a gig in the US state of California.
ROBERTASHARE	Five people have been shot dead in a shooting at a concert in California.
GOLD	Five people have been shot at a California nightclub while Chris Brown was performing.
RND2RND	A council has asked people not to keep their toilets in a bid to save money.
RND2GPT	People are being urged to use a "ladies' toilet" in Skye in Skye in Skye by their own councillor.
BERTSHARE	Complaints about the availability of public toilets on Skye and the isle of Skye is being investigated by highland council.
ROBERTASHARE	Highland council has commissioned a review of public toilets and public toilets on Skye.
GOLD	Islanders on Skye have demanded greater availability of public toilets after complaints some visitors to the Isle are relieving themselves outside.
RND2RND	A man has been jailed for six years for posting offensive comments on Facebook about an Aberdeen teenager who was later found dead.
RND2GPT	A man who admitted killing his six-year-old friend in a disturbance in Aberdeen has been jailed.
BERTSHARE	A man who admitted murdering a toddler after posting offensive comments about him on Facebook has been jailed for three years.
ROBERTASHARE	A man has been jailed for three months for posting "vile" abuse on Facebook about a missing toddler found dead in his Aberdeenshire home.
GOLD	A man who admitted posting offensive comments on Facebook about an Edinburgh boy beaten to death by his mother has been jailed for 12 months.

Figure 1: Model generated and reference summaries used for human evaluation. Words in orange correspond to incorrect or repeated information.

Finally, we estimated the percentage of summaries with at least one repetition of rare or content words. We discarded the 500 most common words from the model generated and reference summaries, the rest were considered as rare or content words. BERTSHARE and ROBERTASHARE summaries improve over the RND2RND summaries, but have more repetitions than the RND2GPT summaries. See examples in Figure 1 for redundant repeated spans marked in orange.

Overall, BERTSHARE and ROBERTASHARE summaries are unequivocally better than RND2GPT summaries in terms of both automatic evaluations (assessing ROUGE) and human evaluations (assessing summary quality); there is still room for improvements in these models (Dong et al., 2019; Song et al., 2019; Lewis et al., 2019).

7 Related Work

Representation Learning. Starting around 2013, word embeddings like word2vec (Mikolov et al., 2013) or GloVe (Pennington et al., 2014) became popular as they were easy to train in an unsupervised fashion on raw text and they improved several downstream tasks when used as features. These word embeddings are invariant to the context in which we the word. There has been previously work to contextualize these embeddings, mainly to account for synonyms (e.g., Huang et al., 2012; Rothe and Schütze, 2015) but only in 2018 did training of the contextualized embeddings using large deep neural networks and an unsupervised training scheme become popular.

Whereas ELMo (Peters et al., 2018) and ULMFiT (Howard and Ruder, 2018) are based on LSTMs (Hochreiter and Schmidhuber, 1997), BERT and GPT are based on the transformer architecture (Vaswani et al., 2017). This architecture outperforms LSTMs on several NLP tasks and we therefore concentrated on these two pre-trained models. The contextualized embedding for each input token is given by the corresponding output of the last encoder layer.

Pre-training Models. One can also see these models as pre-trained models (Dai and Le, 2015), which are then fine-tuned for a downstream task. This is the conceptual view we adopted for this paper. Why unsupervised pre-training helps deep learning was investigated by Erhan et al. (2010). While the unsupervised pre-training strategies are different from those used in our paper, we expect the findings to still hold. They show that unsupervised pre-training is not simply a way of getting a good initial marginal distribution, that classical regularization techniques cannot achieve the same performance as unsupervised pre-training, and that the effect of unsupervised pre-training does not go away with more training data. An extensive study of pre-training was done by Wang et al. (2019a). This study compares single sentence classification, sentence pair classification, seq2seq and language modeling tasks for pre-training, and measures the effect on GLUE. The primary results support the use of language modeling. Peters et al. (2019) explore whether it is preferable to fine-tune the entire model on a specific task or to use the learned representations as features (i.e., freezing the pre-trained model). Their results suggest that the relative performance of fine-tuning vs. feature extraction depends on the similarity between the pre-training and the target tasks. Wang et al. (2019b) propose a combination of both, where first the model is trained with the BERT parameters being frozen and then the entire model is fine-tuned. This is the training scheme we used in the Initializing Only Layers section.

Pre-training for Sequence Generation. Pre-training for seq2seq learning was first done by Ramachandran et al. (2017). They used a language model to pre-train the encoder and decoder of an RNN seq2seq model. Their method improved BLEU scores on newstest2014 by 3 points and ROUGE-L on CNN/DailyMail also by 3 points. However, their BLEU score of 24.7 on newstest2014 En→De, compared to 30.6 in this work, and 29.4 ROUGE-L on CNN/DailyMail, compared with 36.33, also show the superiority of the transformer model as well as the masked language model objective of BERT. MASS (Song et al., 2019) is a BERT-inspired method of pre-training seq2seq models. One advantage of this method is that, in contrast to our setups (except for GPT), the encoder–decoder attention mechanism is also pre-trained. The downside of this approach is that the pre-trained model is task-specific and not as general as BERT or GPT-2. UniLM (Dong et al., 2019) also unifies bidirectional, unidirectional, and seq2seq language modeling. At the time of writing, no public checkpoint was available to us. We compare our work with their results in Table 5. To overcome the issue that the encoder-decoder attention is not pre-trained, Khandelwal et al. (2019) pre-trained a single transformer language model that encodes the source and generates the target. This setup matches our GPT setup. Conneau and Lample (2019) pre-train their model using casual language modeling (like GPT), masked language modeling (like BERT) and a third new objective called translation language modeling to improve cross-lingual pre-training.

Leveraging Public Checkpoints. BERT has been used for various NLP tasks, such as question answering on the SQuAD dataset (Rajpurkar et al., 2018). It also achieved new state-of-the-art results on the GLUE benchmark (Williams et al., 2018) and grounded commonsense inference (SWAG, Zellers et al., 2018). All of these tasks are a form of classification or regression. Liu (2019) fine-tuned BERT for extractive summarization.

An analysis of different layers of the BERT model was performed by Tenney et al. (2019). They found that the classical NLP pipeline appears in the expected sequence. In the context of our experiments in the Initializing a Subset of Layers section, this would mean that the DiscoFuse task profits the most from pre-trained information about POS, constituents, dependencies, and semantic roles. A similar study by Jawahar et al. (2019) found that BERT captures phrase-level information in the lower layers and linguistic information in intermediate layers, with surface features at the bottom, syntactic features in the middle, and semantic features at the top.

GPT was also evaluated on natural language inference tasks. In the extended version of GPT-2,

the model was evaluated on more general natural language processing tasks, like machine translation, reading comprehension, summarization, and language modeling. GPT-2 achieved new state-of-the-art results on several language modeling datasets. On the other tasks, GPT-2 outperformed some unsupervised baselines but is still far behind supervised or task-specific approaches.

After we performed the majority of our experiments, XLNet (Yang et al., 2019), an autoregressive pre-training method based on Transformer XL (Dai et al., 2019), was released. XLNet achieved new state-of-the-art results on several NLP tasks. We leave the experiments with their public checkpoint for future work.

8 Conclusion

We performed an extensive study on leveraging pre-trained checkpoints for sequence generation. Our findings show that a pre-trained encoder is an essential part. Most tasks also profit from sharing the weights between the encoder and the decoder, which additionally decreases the memory footprint. While combing BERT and GPT-2 into a single model often underperformed a randomly initialized baseline, combining RoBERTa and GPT-2 achieved strong results and shows the importance of sharing the vocabulary. Training a language-specific BERT model also improves performance over using the multilingual version.

Acknowledgments

We thank the reviewers and the action editor for their feedback. We would like to thank Ryan McDonald, Joshua Maynez, and Bernd Bohnet for useful discussions.

References

Roee Aharoni and Yoav Goldberg. 2018. Split and rephrase: Better evaluation and stronger baselines. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 719–724. Melbourne, Australia. Association for Computational Linguistics.

Jan A. Botha, Manaal Faruqui, John Alex, Jason Baldridge, and Dipanjan Das. 2018. Learning to split and rephrase from Wikipedia edit history. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 732–737. Brussels, Belgium. Association for Computational Linguistics.

William Chan, Nikita Kitaev, Kelvin Guu, Mitchell Stern, and Jakob Uszkoreit. 2019. KERMIT: Generative insertion-based modeling for sequences. *CoRR*, abs/1906.01604v1.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining, In H. Wallach, H. Larochelle, A. Beygelzimer, F. Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 7057–7067. Curran Associates, Inc.

Andrew M. Dai and Quoc V. Le. 2015. Semi-supervised sequence learning, In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3079–3087. Curran Associates, Inc.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

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

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In H. Wallach, H. Larochelle, A. Beygelzimer, F. Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 13042–13054. Curran Associates, Inc.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. 2010. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11:625–660.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 1243–1252. Sydney, Australia.

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.

Mor Geva, Eric Malmi, Idan Szpektor, and Jonathan Berant. 2019. DiscoFuse: A large-scale dataset for discourse-based sentence fusion. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3443–3455, Minneapolis, Minnesota. Association for Computational Linguistics.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1631–1640, Berlin, Germany. Association for Computational Linguistics.

Nikolaus Hansen. 2016. The CMA evolution strategy: A tutorial. *CoRR*, abs/1604.00772.

Tianyu He, Xu Tan, Yingce Xia, Di He, Tao Qin, Zhibo Chen, and Tie-Yan Liu. 2018. Layer-wise coordination between encoder and decoder for neural machine translation. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 7944–7954, Curran Associates, Inc.

Dan Hendrycks and Kevin Gimpel. 2016. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *CoRR*, abs/1606.08415.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1693–1701. Curran Associates, Inc.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Eric Huang, Richard Socher, Christopher Manning, and Andrew Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 873–882, Jeju Island, Korea. Association for Computational Linguistics.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Urvashi Khandelwal, Kevin Clark, Dan Jurafsky, and Lukasz Kaiser. 2019. Sample efficient text summarization using a single pre-trained transformer. *CoRR*, abs/1905.08836.

Byeongchang Kim, Hyunwoo Kim, and Gunhee Kim. 2019. Abstractive summarization of

Reddit posts with multi-level memory networks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2519–2531. Minneapolis, Minnesota. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461.

Chin Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 150–157.

Yang Liu. 2019. Fine-tune BERT for extractive summarization. *CoRR*, abs/1903.10318.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Jordan J. Louviere, Terry N. Flynn, and Anthony Alfred John Marley. 2015. *Best-worst scaling: Theory, methods and applications*. Cambridge University Press.

Jordan J. Louviere and George G. Woodworth. 1991. Best-worst scaling: A model for the largest difference judgments. *University of Alberta*, Working Paper.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Courtney Napoles, Matthew Gormley, and Benjamin Van Durme. 2012. Annotated gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, pages 95–100, Montreal, Canada. Association for Computational Linguistics.

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

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018b. Ranking sentences for extractive summarization with reinforcement learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1747–1759, New Orleans, Louisiana. Association for Computational Linguistics.

Shashi Narayan, Claire Gardent, Shay B. Cohen, and Anastasia Shimorina. 2017. Split and rephrase. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 606–616. Copenhagen, Denmark. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543. Doha, Qatar. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2227–2237. New Orleans, Louisiana. Association for Computational Linguistics.

Matthew E. Peters, Sebastian Ruder, and Noah A. Smith. 2019. To tune or not to tune? adapting pretrained representations to diverse tasks. In *Proceedings of the 4th Workshop on Representation Learning for NLP*, pages 7–14. Florence, Italy. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *Technical report, OpenAI*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *Technical report, OpenAI*.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Prajit Ramachandran, Peter Liu, and Quoc Le. 2017. Unsupervised pretraining for sequence to sequence learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 383–391. Copenhagen, Denmark. Association for Computational Linguistics.

Sascha Rothe and Hinrich Schütze. 2015. AutoExtend: Extending word embeddings to embeddings for synsets and lexemes. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 1793–1803, Beijing, China. Association for Computational Linguistics.

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

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 464–468, New Orleans, Louisiana. Association for Computational Linguistics.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. MASS: Masked sequence to sequence pre-training for language generation. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pages 5926–5936. PMLR.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pages 5998–6008.

Alex Wang, Jan Hula, Patrick Xia, Raghavendra Pappagari, R. Thomas McCoy, Roma Patel, Najoung Kim, Ian Tenney, Yinghui Huang, Katherin Yu, Shuning Jin, Berlin Chen, Benjamin Van Durme, Edouard Grave, Ellie Pavlick, and Samuel R. Bowman. 2019a. Can you tell me how to get past sesame street? Sentence-level pretraining beyond language modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4465–4476, Association for Computational Linguistics.

Ran Wang, Haibo Su, Chunye Wang, Kailin Ji, and Jupeng Ding. 2019b. To tune or not to tune? how about the best of both worlds? *CoRR*, abs/1907.05338.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through

inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. XLNet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 93–104, Brussels, Belgium. Association for Computational Linguistics.