BART

This article reviews a popular paper written by a team at Facebook, which proposes a more generalized approach to pre-training models for language processing. This forms a base of an approach we will be building on in our project, and it is useful to bring myself up to speed on NLP research to be better equipped to work on our project.

Introduction

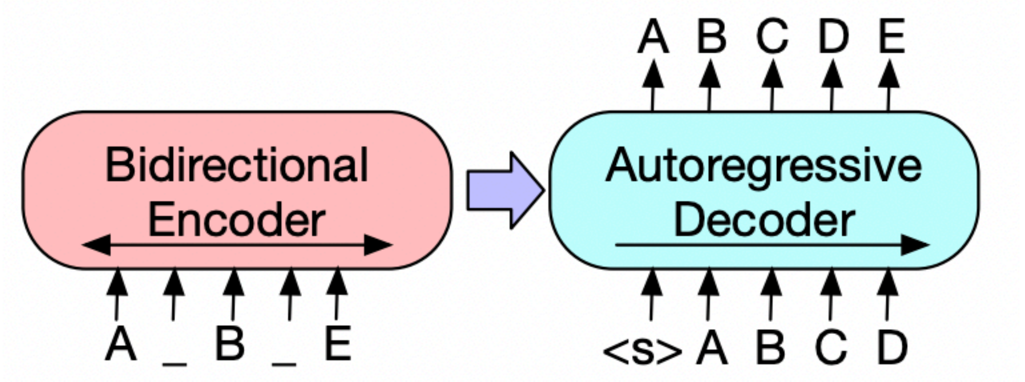
BART is a denoising encoder for pre-training sequence to sequence models, which uses a standard transformer based neural machine translation architecture. It proposes a pre-training approach which corrupts the text with a set of defined noising functions, and learning a model to reconstruct the original text. BART can be seen as a generalization of BERT, GPT, and other pre-training schemes, and is particularly effective for text generation, where it has achieved state of the art results.

BART's Architecture

The model is a standard sequence to sequence transformer, which contains two major components:

- A bidirectional encoder, similar to BERT, which "learns" what the sentence means, and stores it in a hidden state.
- An autoregressive decoder, to predict the next token, given a previous sequence of tokens. The state of the encoder's final hidden state is fed into the decoder, and is used when producing the necessary output

The below image represents a simplified view of the architecture, with an input corrupted by "token masking" as defined below, and the expected output produced by the decoder.



Bart's Training

BART's training consists of two stages, pre-training and fine-tuning. During pre-training, the model is trained on a large unlabelled dataset, during which it learns to denoise corrupted text (terms defined in the next section). The pre-trained models are then used as the initial states for the fine-tuning tasks listed below, which can better handle each specific task. We look into each of these components in the following sections.

Corruption Transformations for Pre-training

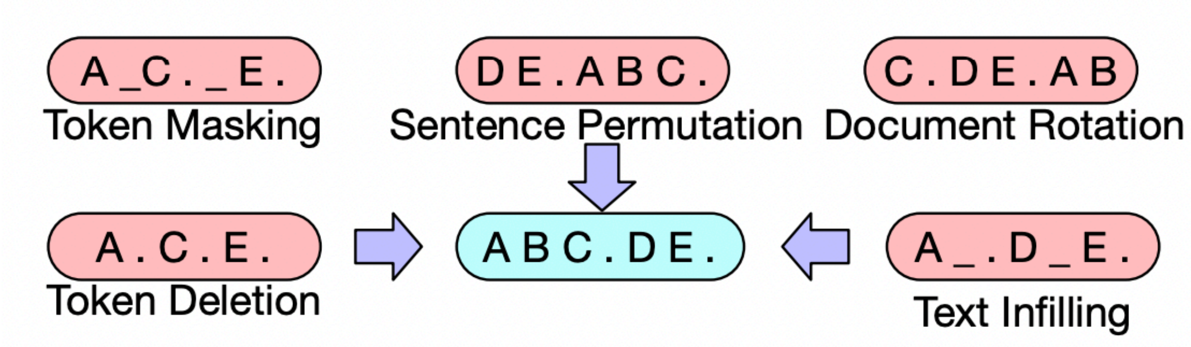
The model is pre-trained to predict masked tokens, in a two step process:

- "Corrupting" portions of the input, by transforming the input text in a randomly selected method picked from the approaches listed below.
- "Denoising" the sequence by predicting the original tokens for each masked token.

The following corruption transformations are compared in the paper:

1. Token Masking: Tokens in the input are randomly sampled, and masked, and the neural network needs to predict the masked word. This is the primary approach used by BERT (80% of the time based on the original paper).
2. Token Deletion: Tokens in the input are randomly deleted, and the neural network needs to predict the locations of deleted words, alongside the masked word itself.
3. Token Infilling: A collection of text spans are sampled (with the span length governed by a Poisson distribution), and replaced with a single masked token. The neural network will need to predict an entire masked sequence, without knowing the number of masked tokens.
4. Sentence Permutation. The sentences in the document are shuffled, and the neural network needs to reconstruct the original order.
5. Document Rotation: The document is rotated about some random token, and the model needs to identify the start of the document.

These 5 transformations, and the target end state can be graphically summarized by the following image:



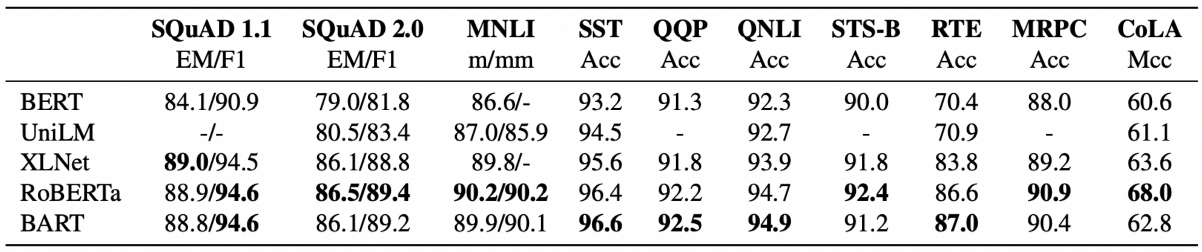
Fine Tuning BART

The representations produced during pre-training can then be trained in different ways, based on our goals. The paper discusses approaches for handling sequence classification tasks, token classification tasks, sequence generation tasks and for machine translation.

We are specifically interested in the sequence generation task, which we will leverage in our project. In this case, BART's autoregressive decoder can be fine tuned for specific sequence generation tasks, such as creating an abstract or for answering user questions. The architecture needs no changes to handle this, and the encoder can be fed a body of text, and the decoder autoregressively generates the output. This output can then be constrained as

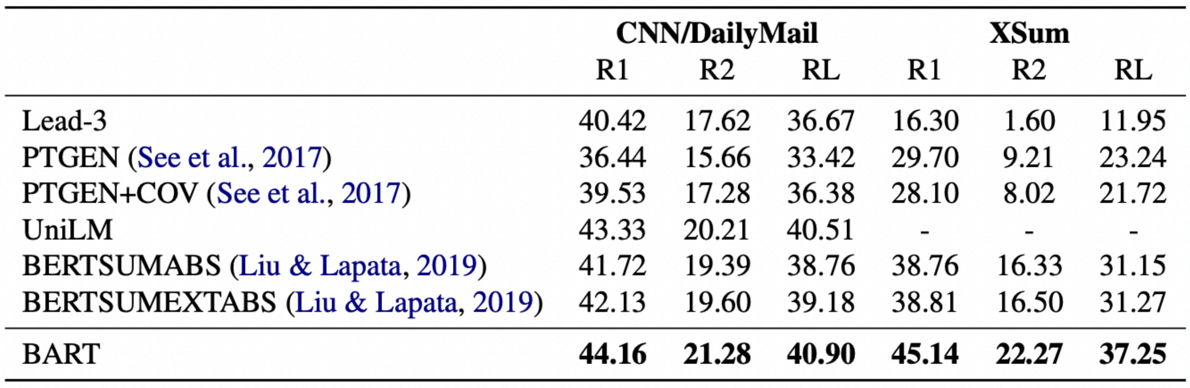
Comparison with other Approaches

Discriminative Tasks: The below picture has been taken from the result paper, comparing the results seen on the SQuAD and GLUE tasks, where BART is seen to outperform BERT for the various discriminative tasks, and performs similar to RoBERTa and XLNet.



	SQuAD 1.1	**SQuAD 2.0**	**MNLI**	**SST**	**QQP**	**QNLI**	**STS-B**	**RTE**	**MRPC**	**CoLA**
	EM/F1	EM/F1	m/mm	Acc	Acc	Acc	Acc	Acc	Acc	Mcc
BERT	84.1/90.9	79.0/81.8	86.6/-	93.2	91.3	92.3	90.0	70.4	88.0	60.6
UniLM	-/-	80.5/83.4	87.0/85.9	94.5	-	92.7	-	70.9	-	61.1
XLNet	**89.0**/94.5	86.1/88.8	89.8/-	95.6	91.8	93.9	91.8	83.8	89.2	63.6
RoBERTa	88.9/**94.6**	**86.5/89.4**	**90.2/90.2**	96.4	92.2	94.7	**92.4**	86.6	**90.9**	**68.0**
BART	88.8/**94.6**	86.1/89.2	89.9/90.1	**96.6**	**92.5**	**94.9**	91.2	**87.0**	90.4	62.8

Summarization: BART really shines in summarization tasks, and provides a significant improvement over BERT based models for the XSum task in particular. This occurs because XSum's summaries don't entirely resemble the source sentences, which make it harder for regular extractive models to perform well.



	CNN/DailyMail			**XSum**		
	R1	R2	RL	R1	R2	RL
Lead-3	40.42	17.62	36.67	16.30	1.60	11.95
PTGEN (See et al., 2017)	36.44	15.66	33.42	29.70	9.21	23.24
PTGEN+COV (See et al., 2017)	39.53	17.28	36.38	28.10	8.02	21.72
UniLM	43.33	20.21	40.51	-	-	-
BERTSUMABS (Liu & Lapata, 2019)	41.72	19.39	38.76	38.76	16.33	31.15
BERTSUMEXTABS (Liu & Lapata, 2019)	42.13	19.60	39.18	38.81	16.50	31.27
BART	**44.16**	**21.28**	**40.90**	**45.14**	**22.27**	**37.25**

BART also performs well for other tasks, but the results are omitted for brevity.

Comparison of BART and BERT

- BERT only contains the first component of the architecture proposed by BART, and operates exclusively with the bidirectional component, and does not include the autoregressive decoder.
- BART uses a wider variety of corruption transforms when compared to BERT, which lets the model infer a deeper level of meaning going beyond masking of specific words which constitutes 80% of BERT's corruption method. However, BERT uses two approaches not covered in here, substituting a random word, and not performing any changes, which would be interesting corruption approaches for BART.
- Neural Network Architecture
  - Unlike BERT, each layer in BART's decoder performs cross-attention over the last hidden layer of the bidirectional encoder.
  - BERT uses a feed-forward network before word prediction, which is not used in BART
  - The net effect of these two changes is that a BART model contains ~10% more parameters when compared to an equivalent BERT model.

Conclusion

- BART proposes a state of the art pre-training method for text generation tasks, and performs on par with other existing methods for discriminative tasks.
- It serves as a generalization of other approaches, and provides a framework into which future approaches can plug into for handling more specific tasks.
- A pre-trained model has been made available for use in pytorch, which allows users to quickly get started on building on top of BART.

References

1. Lewis, Mike, et al. "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension." *arXiv preprint arXiv:1910.13461* (2019).
2. Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).
3. Stanford's NLP course for a grounding in this field: https://web.stanford.edu/class/cs224n/index.html
4. https://towardsdatascience.com/bart-for-paraphrasing-with-simple-transformers-7c9ea3dfdd8c
5. https://medium.com/analytics-vidhya/revealing-bart-a-denoising-objective-for-pretraining-c6e8f8009564