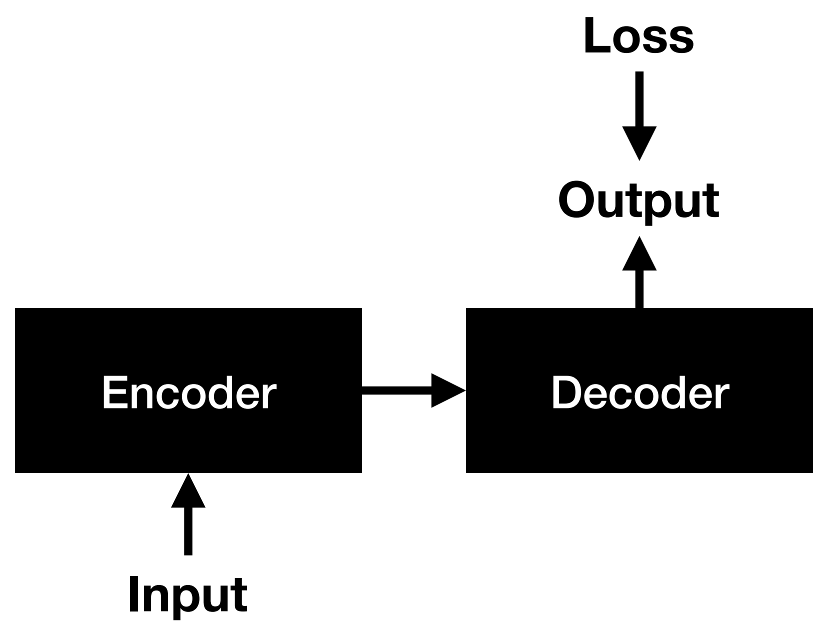
Raw Data

Pre-training Encoder-Decoders

Cornell CS 5740: Natural Language Processing
Yoav Artzi, Spring 2023

Encoder-decoder

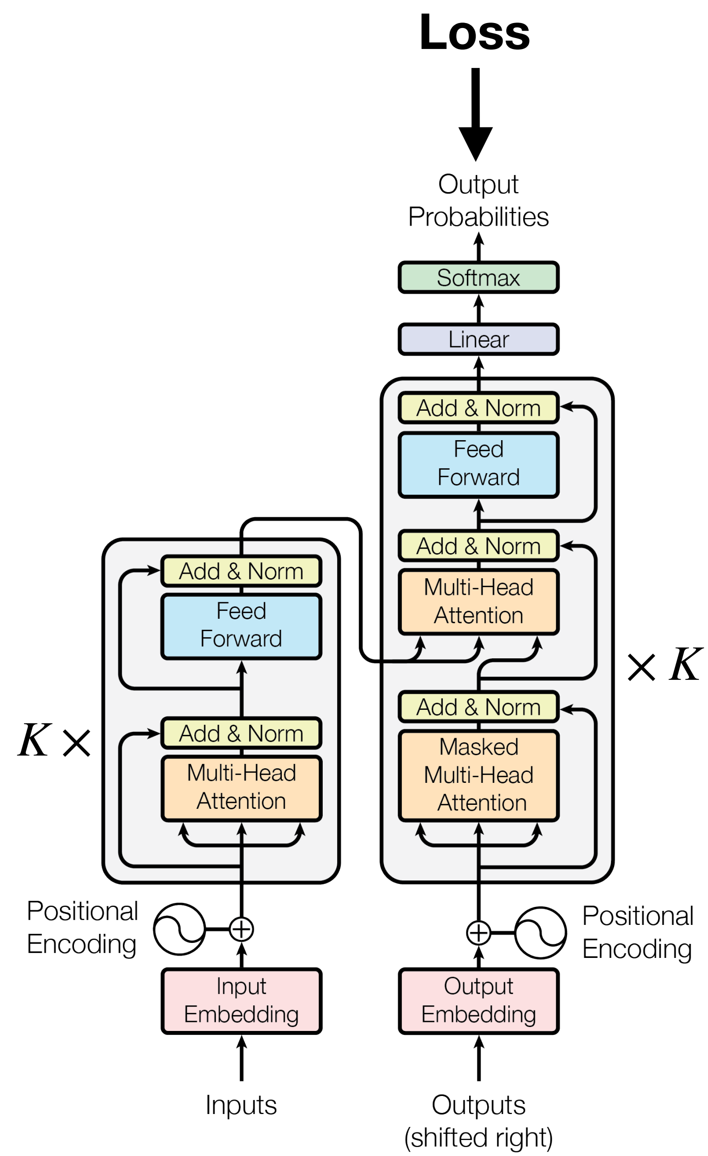
- The model is composed of two components
- Bidirectional **encoder** to process the input
- Autoregressive **decoder** to generate output
- Training is usually done with loss on the output
  - Propagates into the decoder and through it to the encoder



Encoder-decoder

With Transformers

- The model is composed of two components
- Bidirectional **encoder** to process the input
- Autoregressive **decoder** to generate output
- Training is usually done with loss on the output
    - Propagates into the decoder and through it to the encoder

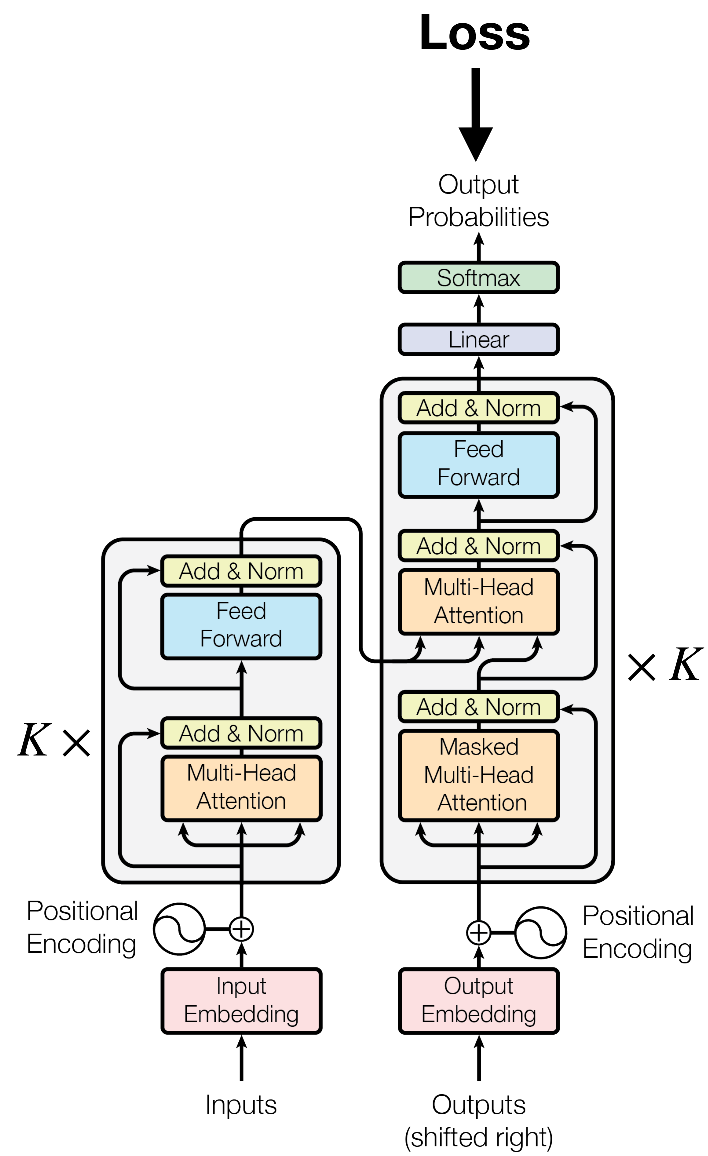


[Vaswani et al. 2017]

Encoder-decoder

With Transformers

- Bidirectional **encoder** to process the input
- Autoregressive **decoder** to generate output
- Why does this structure make sense?



[Vaswani et al. 2017]

Encoder-decoder Pre-training

The BART Recipe

- An encoder-decoder (sequence-to-sequence) pre-trained model
- Extends the BERT approach to encoder-decoder

[Lewis et al. 2019]

BART

BERT Reminder

- Encoder-only Transformer
- Trained on raw data
- Two self-supervised objects:
  - Masked LM
  - Next-sentence prediction
- Transformed the NLP task landscape — if you have enough data, fine-tuned BERT works really well

[Lewis et al. 2019]

BART

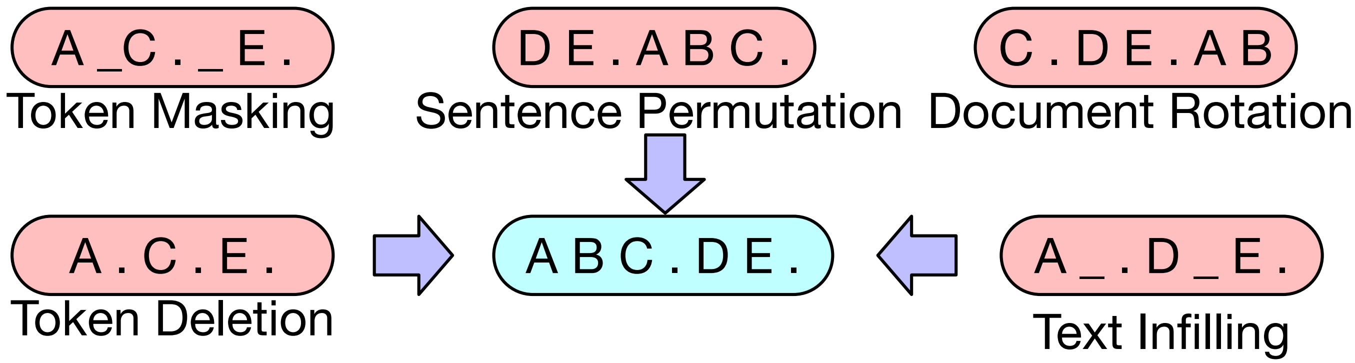
- How does the BERT learning approach adapts to an encoder-decoder architecture?
  - Output is generated by decoder, and the loss is on the output
  - Input is a sequence of tokens

[Lewis et al. 2019]

BART

Denoising Self-supervised Objective

- Corrupt the input following five different recipes



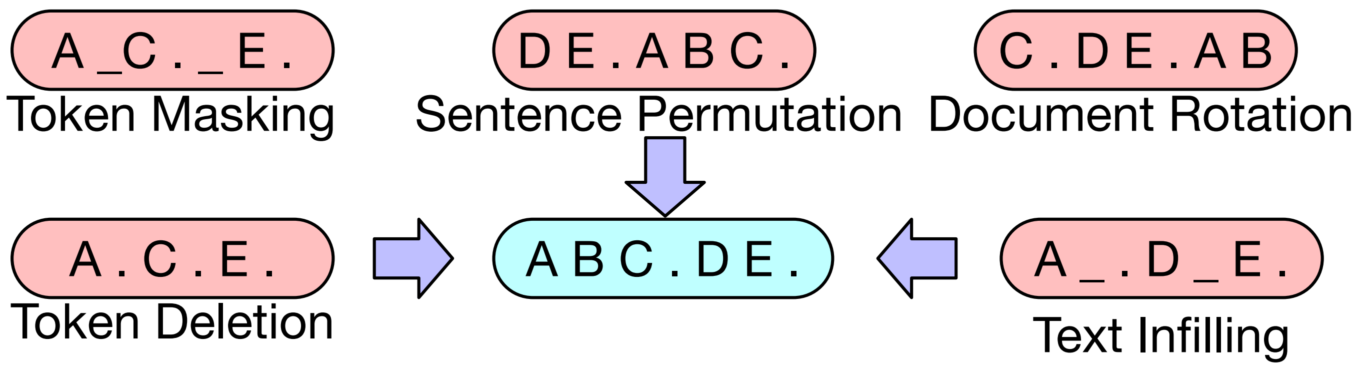
- Try to recover the pre-corrupted input by generating it using the decoder
- Train on a lot of raw text data, just like with BERT
- How to compute the loss? Loss can be computed using “teacher forcing”

[Lewis et al. 2019]

BART

Denoising Self-supervised Objective

- Corrupt the input following five different recipes



- Try to recover the pre-corrupted input by generating it using the decoder

- How to compute the loss? Loss can be computed using “teacher forcing”

[Lewis et al. 2019]

BART

What Do We Get?

- BERT: a pre-trained encoder
- BART: pre-trained decoder and encoder
  - Can use both
  - Or can use only the decoder wherever we would use BERT

[Lewis et al. 2019]

BART

How to Use?

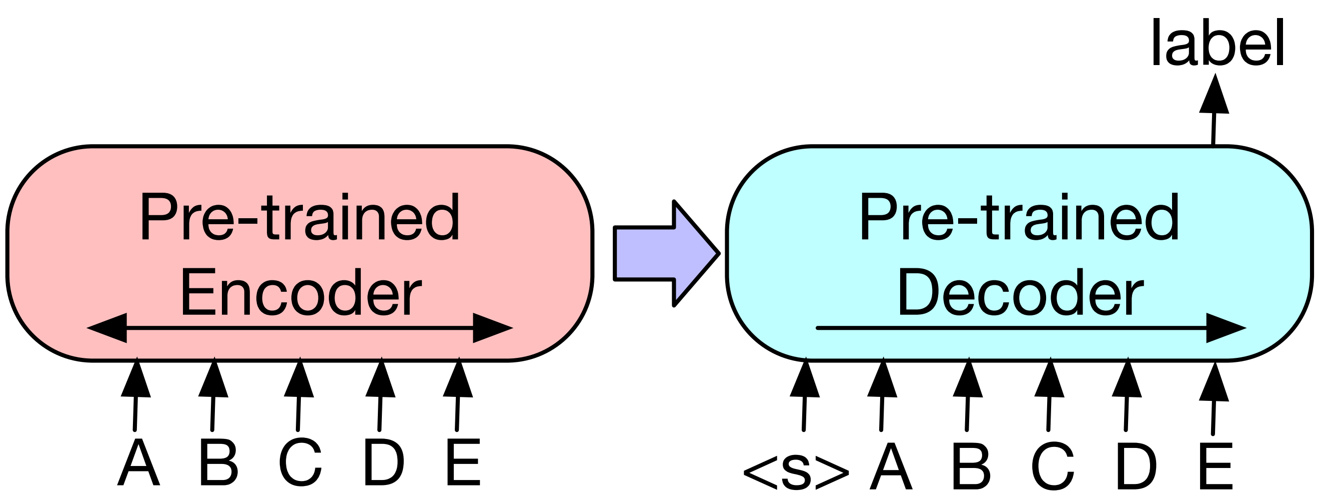
- Similar to BERT: fine-tune for the end task
- Very natural for summarization
  - Because input and output vocabulary are the same
- How can we use for classification
- What about machine translation?

[Lewis et al. 2019]

BART

Classification

- The input is given the encoder
- The same input is forced decoded in the decoder via “teacher forcing”
- The representation from the final decoder hidden state is given to a classification head



[Lewis et al. 2019]

BART

Machine Translation

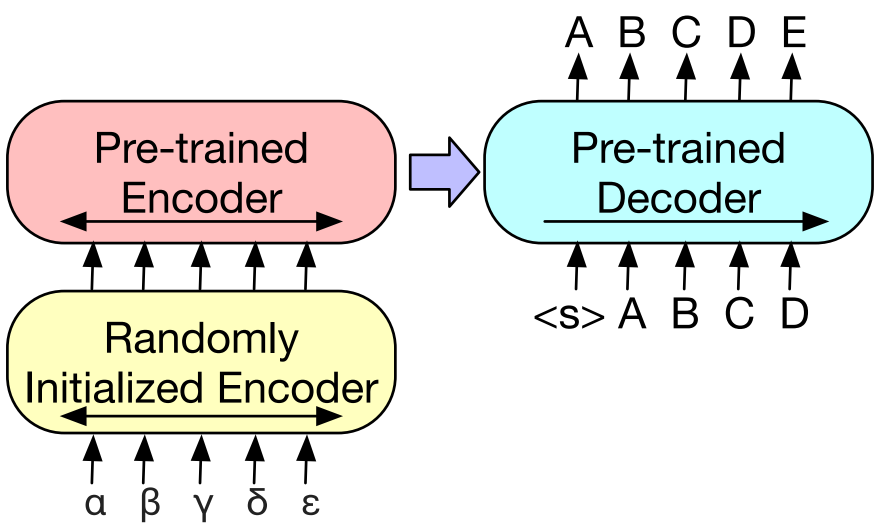
- In MT, the input and output vocabularies are different
- When is that a problem with BART?
- How can we solve it?

[Lewis et al. 2019]

BART

Machine Translation

- In MT, the input and output vocabularies are different
- When is that a problem with BART?
- How can we solve it?
- Add a small pre-encoder encoder to replace the BART input embeddings with computed embeddings



[Lewis et al. 2019]

BART

Performance

- Can do anything that BERT does
- But can also do generation tasks (e.g., summarization)

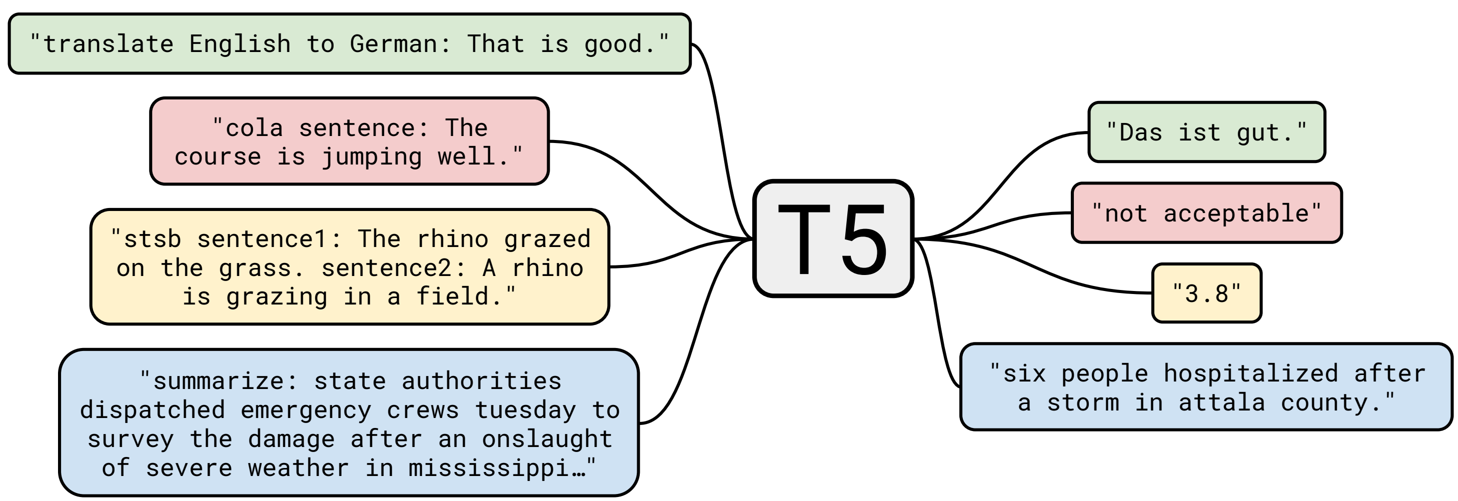
Model	SQuAD 1.1 F1	MNLI Acc	ELI5 PPL	XSum PPL	ConvAI2 PPL	CNN/DM PPL
BERT Base (Devlin et al., 2019)	88.5	**84.3**	-	-	-	-
BART Base						
w/ Token Masking	90.4	84.1	25.05	7.08	11.73	6.10
w/ Token Deletion	90.4	84.1	24.61	6.90	11.46	5.87
w/ Text Infilling	**90.8**	84.0	24.26	**6.61**	**11.05**	5.83
w/ Document Rotation	77.2	75.3	53.69	17.14	19.87	10.59
w/ Sentence Shuffling	85.4	81.5	41.87	10.93	16.67	7.89
w/ Text Infilling + Sentence Shuffling	**90.8**	83.8	24.17	6.62	11.12	**5.41**

[Lewis et al. 2019]

Encoder-decoder Pre-training

The T5 Recipe

- Concurrent and similar to BART
- Also adopted the text-to-text approach for all NLP tasks



[Raffel et al. 2019]

T5

Pretraining

- Pretraining is similar to the denoising objective of BART:
  - Input: text with gaps (phrases removed)
  - Output: sequence of phrases to fill the gaps

Original text

Thank you ~~for inviting~~ me to your party ~~last~~ week.

Inputs

Thank you <X> me to your party <Y> week.

Targets

<X> for inviting <Y> last <Z>

[Raffel et al. 2019]

T5

Results

- T5 was trained on one of the first very large corpora: 750GB of text, with pre-training using 2^{35} tokens
- First to show the impact of data scale
- Why did they repeat the smaller scale datasets?

Number of tokens	Repeats	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Full data set	0	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
2^{29}	64	**82.87**	**19.19**	**80.97**	**72.03**	**26.83**	**39.74**	**27.63**
2^{27}	256	82.62	**19.20**	79.78	69.97	**27.02**	**39.71**	27.33
2^{25}	1,024	79.55	18.57	76.27	64.76	26.38	39.56	26.80
2^{23}	4,096	76.34	18.33	70.92	59.29	26.37	38.84	25.81

[Raffel et al. 2019]

BART and T5

Takeaways

- BART and T5 are very useful for all sorts of sequence-to-sequence tasks with language
  - T5 comes in different sizes
  - There are various customization (e.g., CodeT5)
- Extended the generalizations conclusions from BERT, and demonstrated the impact of data scale

Acknowledgements

- Slides are based on slides from UT Austin CS 388 by Greg Durrett