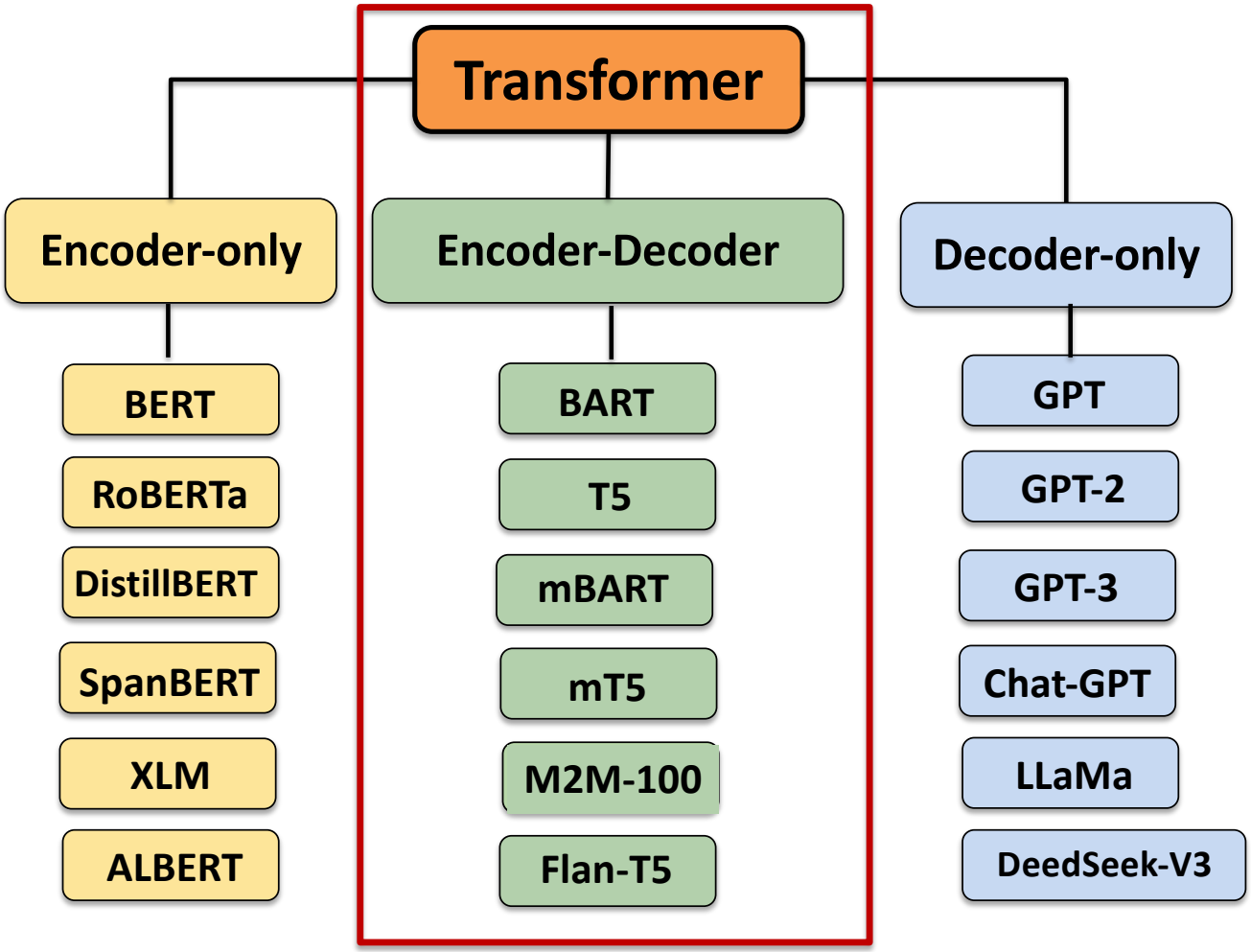
BART and T5

AI with Deep Learning
EE4016

Prof. Lai-Man Po
Department of Electrical Engineering
City University of Hong Kong

Well-known Transformer-based Models



Transformer-based Language Models

- **Encoder-only Transformer Models**
  - **Masked Language Models:** Designed to consider bidirectional contexts that are beneficial to classification and language understanding
  - **BERT Series**: **BERT** (2018-10), **DistilBERT**, **RoBERTa**, **XLM**, **XLM-RoBERTa**, **ALBERT**, **ELECTRA**
- **Decoder-only Transformer Models**
  - **Autoregressive Language Models:** Designed to predict the next token (sub-word), suitable for text generation
  - **GPT-1** (2018-6), **GPT-2** (2019-02), **GPT-**3 (2020-06), **InstructGPT** (2022-03), **ChatGPT** (2022-11), **GPT-4** (2023-03), **GPT-4o** (2024-04)
- **Encoder-Decoder Transformer Models**
  - **Sequence-to-sequence models**
  - Original **Transformer** (2017-06), **T5** (2019-10), **BART** (2019-11), **M2M-100** (2021-07), **Flan-T5** (2022-10)

BART

Bidirectional Auto-Regressive Transformers

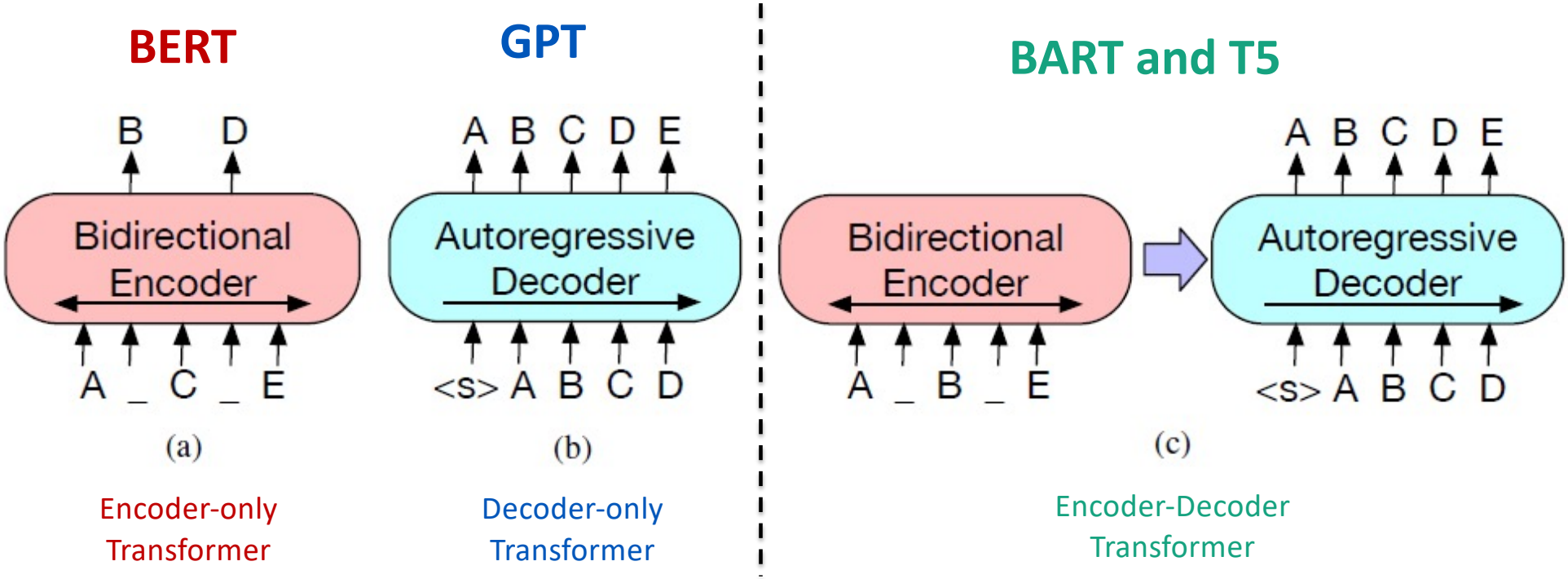
Meta AI's BART (2019-10)

- **BART** combines the pre-training objectives of Google's **BERT** and OpenAI's **GPT** within the **Encoder-Decoder Transformer Architecture**.
  - **BERT**'s **bidirectional, autoencoder nature** is good for classification downstream tasks that require information about the entire sequence.
  - **BERT** is **not that effective for generation tasks** where generated words should only depend on previously generated words.
  - **GPT**'s **unidirectional autoregressive nature** is beneficial to text generation.
  - **GPT** is **less suitable for tasks that require information about the entire sequence**, such as classification.
- **BART** combines the strengths of both BERT and GPT, making it the best of both worlds.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., & Zettlemoyer, L. (2019). BART: Denoising Sequence-to- Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. http://arxiv.org/abs/1910.13461

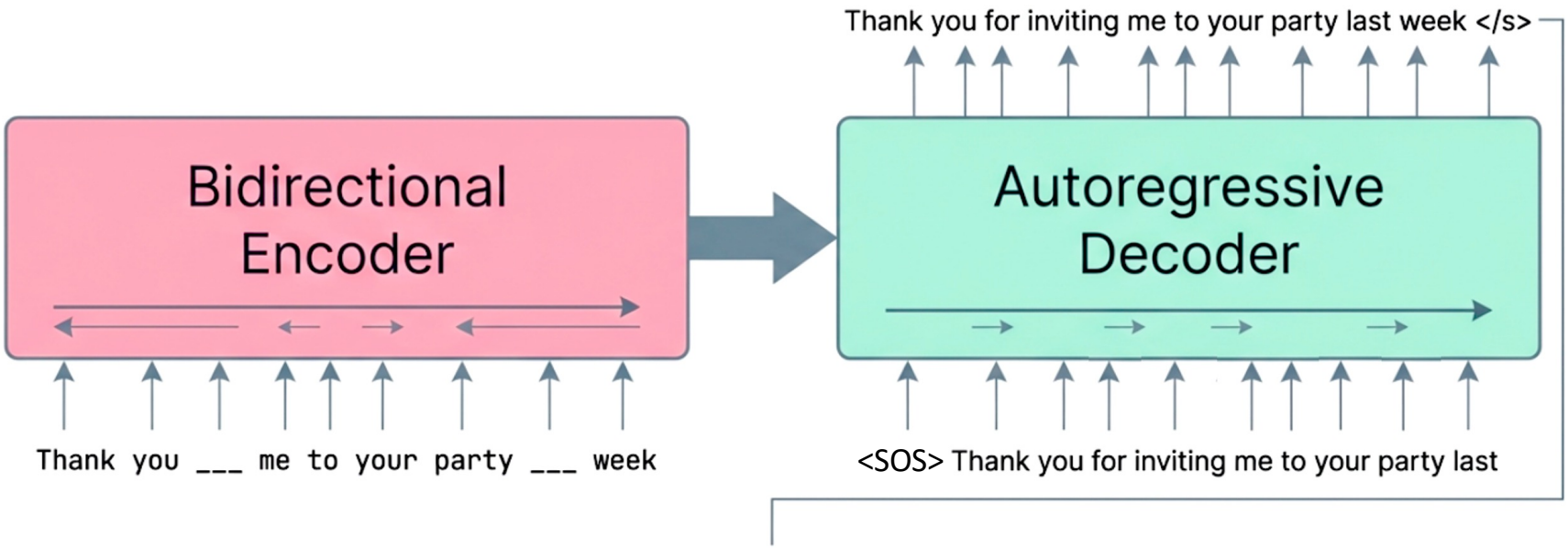
BART vs BERT and GPT

- **BART** = **BERT** Encoder + **GPT** Decoder + **Noise Transformations**



BART: The Denoising Autoencoder

Bidirectional and Auto-Regressive Transformers



Key Mechanism: BART predicts the **whole sentence** (reconstructing the original input).

Pre-training: Learning from Noise

BART learns by reconstructing corrupted text

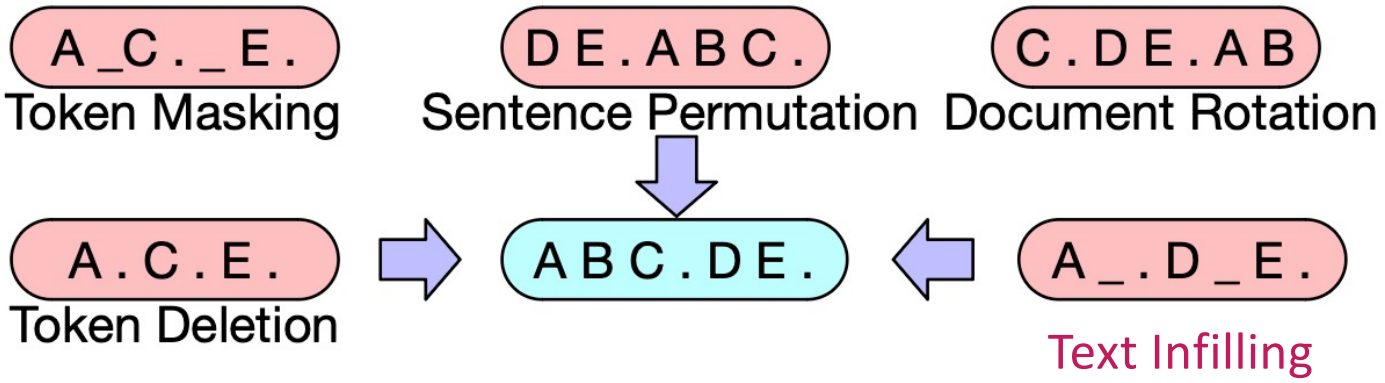


Figure 2: Transformations for noising the input that we experiment with. These transformations can be composed.

BART: Noise Transformations

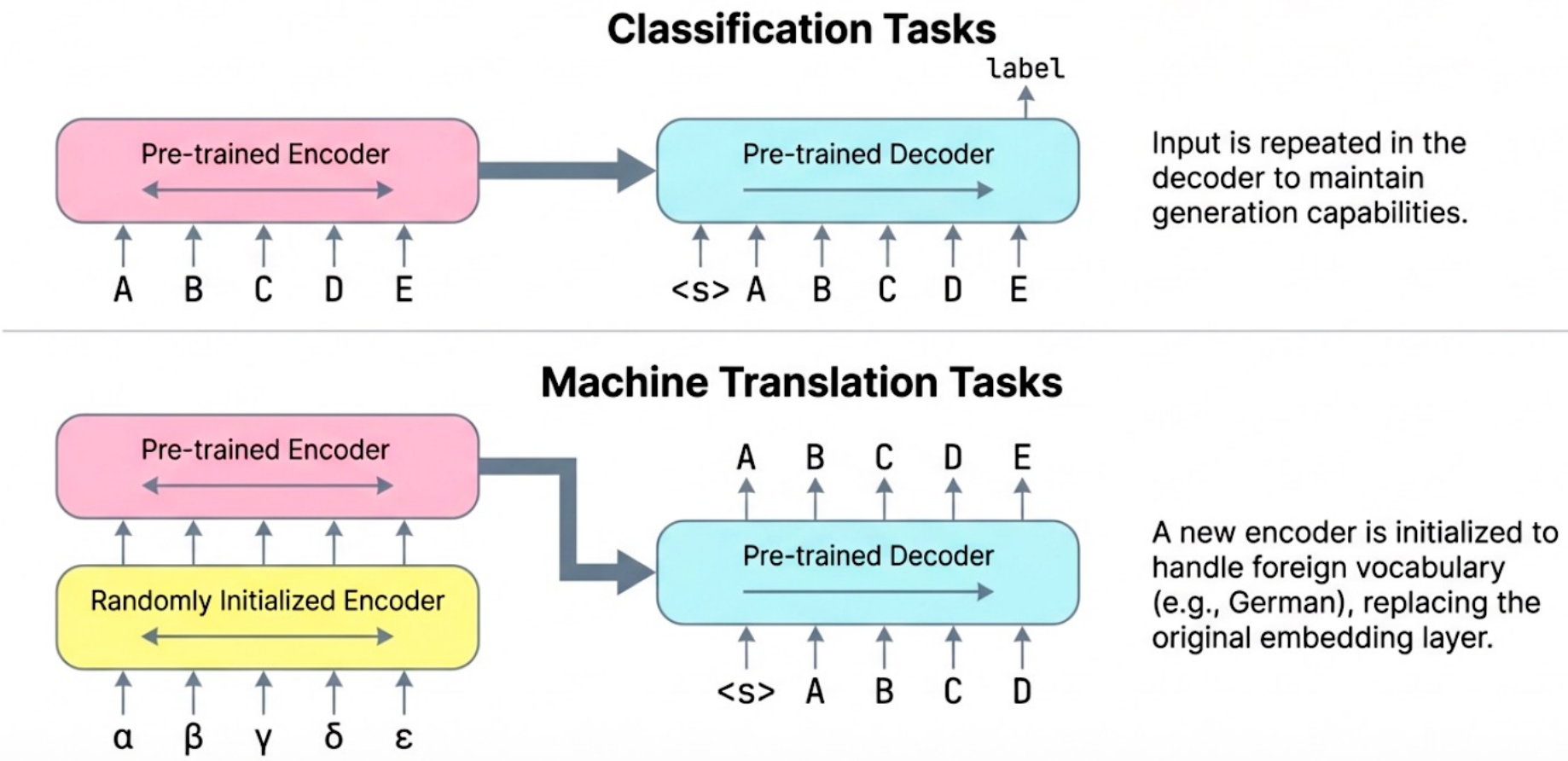
- **Token Masking**: Same as BERT, random tokens are sampled and replaced with [MASK] elements.
- **Token Deletion**: Random tokens are deleted from the input. The model must decide which positions are missing inputs.
- **Sentence Permutation**: A document is divided into sentences based on full stops, and these sentences are shuffled in a random order.
- **Document Rotation**: The document is rotated so that it begins with that token. This task trains the model to identify the start of the document.
- **Text Infilling**: Text infilling is based on SpanBERT. We **replace samples of different lengths with [MASK] tokens of the same length**. This trains the model **to predict missing tokens in a span**.

BART: Comparison of Pre-Training Objectives

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**	-	-	-	-
Masked Language Model	90.0	83.5	24.77	7.87	12.59	7.06
Masked Seq2seq	87.0	82.1	23.40	6.80	11.43	6.19
Language Model	76.7	80.1	**21.40**	7.00	11.51	6.56
Permuted Language Model	89.1	83.7	24.03	7.69	12.23	6.96
Multitask Masked Language Model	89.2	82.4	23.73	7.50	12.39	6.74
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**

Except ELI5, BART models using text-infilling perform well on all tasks.

BART: Fine-Tuning Strategies



BART Performance Benchmarks

Table 1. **Discriminative Tasks (SQuAD & GLUE)**

Model	SQuAD 1.1 F1	SST Acc	MNLI Acc
BERT	90.9	93.2	86.6
RoBERTa	94.6	96.4	90.2
BART	**94.6**	**96.6**	90.1

Comparable to RoBERTa on understanding tasks.

Table 2. **Generative Tasks (Summarization)**

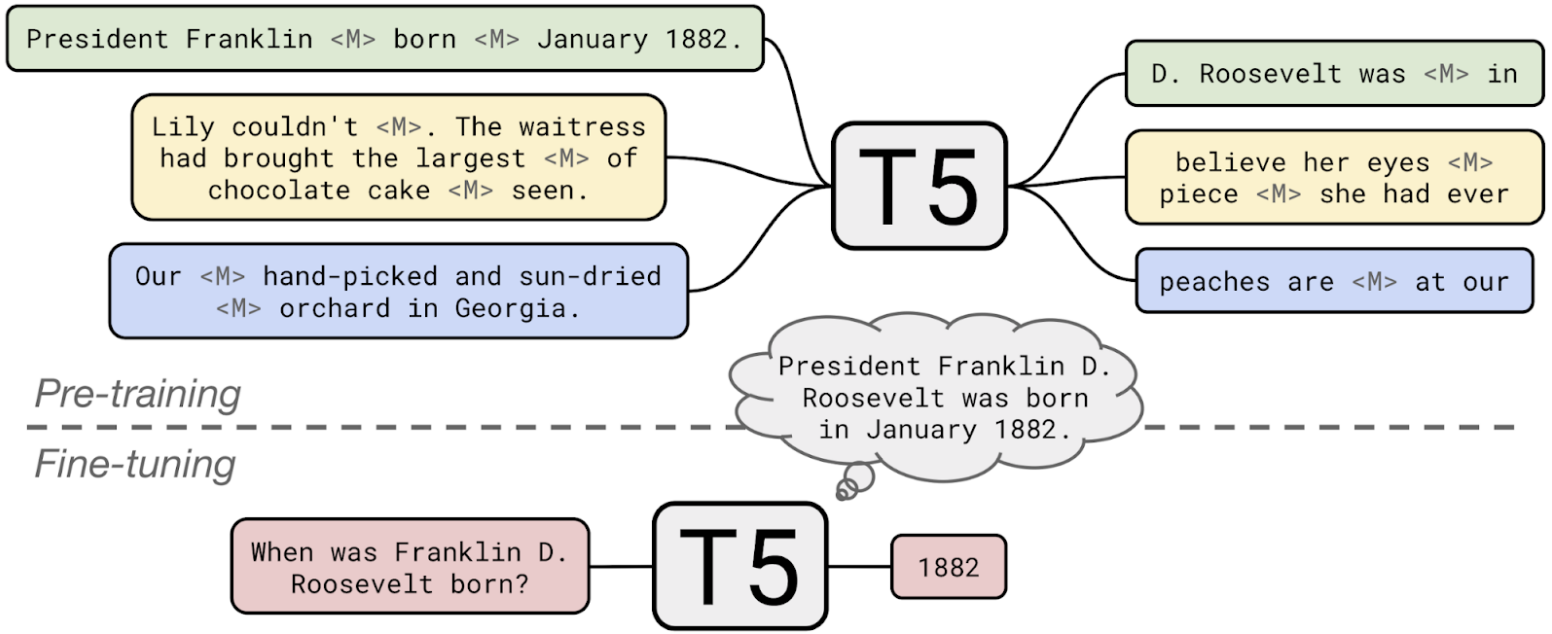
Dataset	Metric	BART Score	Previous Best
CNN/DailyMail	ROUGE-1	**44.16**	41.72
XSum	ROUGE-1	**45.14**	38.81
ELI5 (QA)	ROUGE-1	**30.6**	28.9

State-of-the-art results on Summarization and Abstractive QA.

Text-To-Text Transfer Transformer

Google T5 (2019-11): The Text-to-Text Transformer

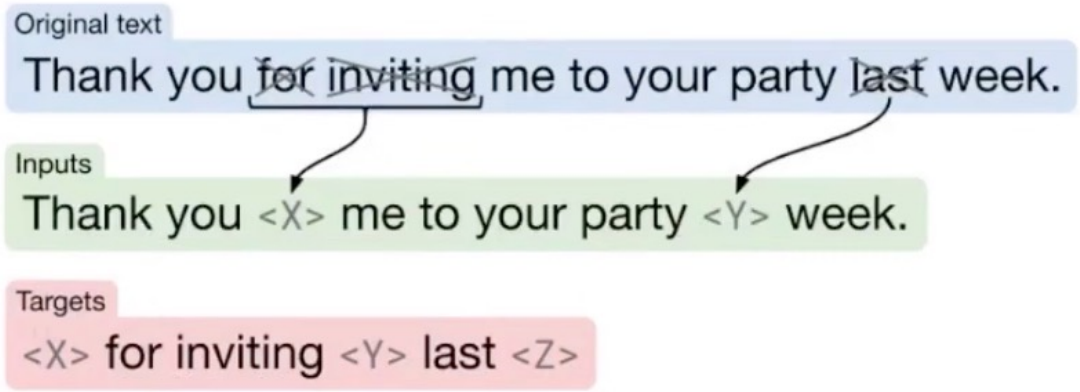
Unified Framework: Everything is Text-to-Text



Raffel, Colin, et al. "Exploring the limits of transfer learning with a unified text-to-text transformer." J. Mach . Learn. Res. 21.140 (2020): 1-67

T5 Training Objective

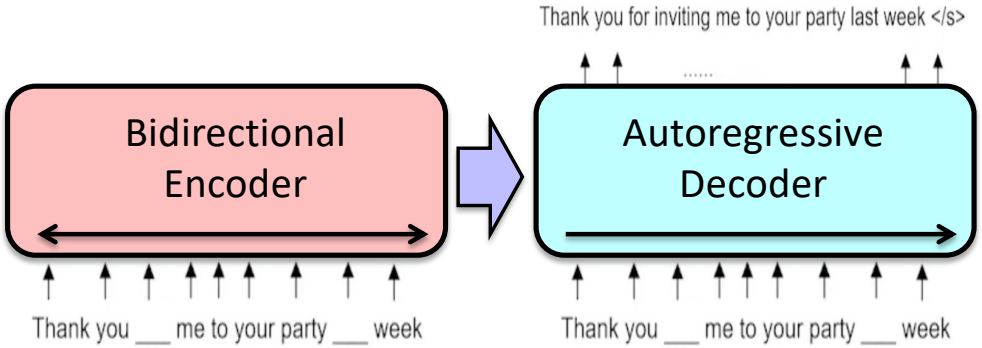
- **Dataset**: **C**olossal **C**lean **C**rawled **C**orpus (**C4**) dataset, a cleaned version of Common Crawl (deduplication, discarding incomplete sentences, and removing offensive or noisy content)
- **Pre-training**: Self-supervised to **predict** the masked tokens **only**



T5 vs BART: Pretraining

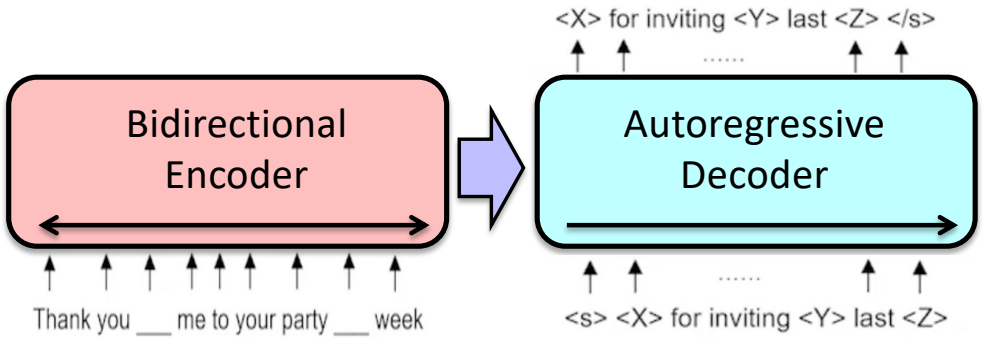
- T5 is **similar** to BART but its denoising pre-training is different.

BART



Predicts the **whole sentence**
(reconstruct the the sentence)

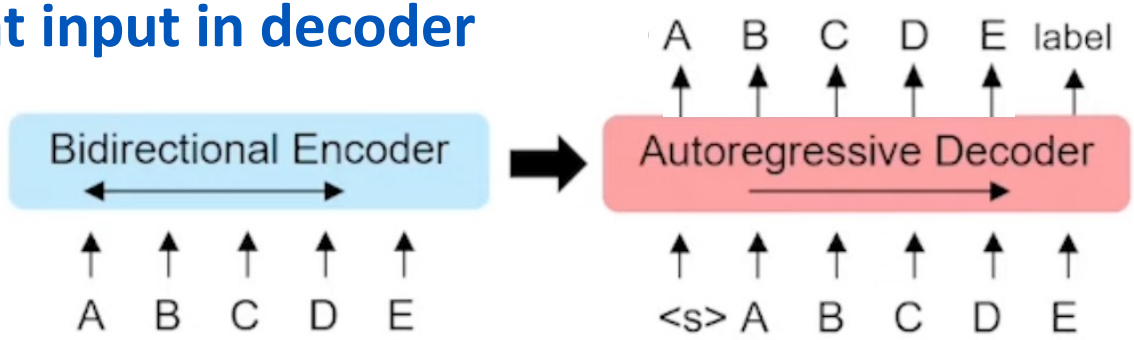
T5



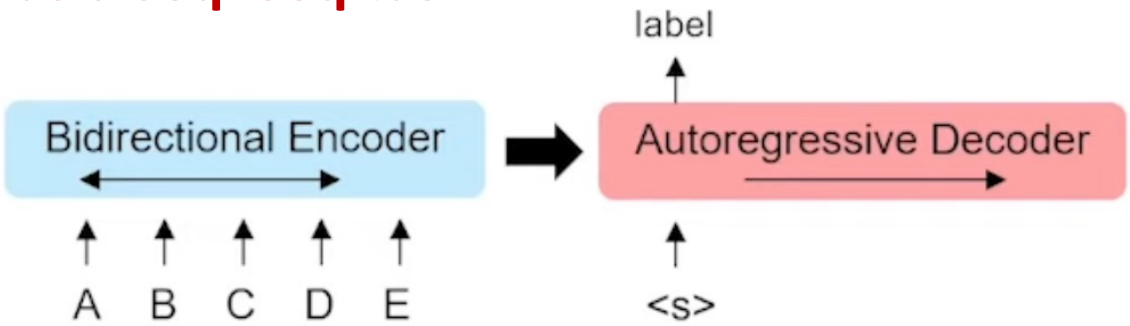
T5: Predicts the **missing parts only**

BART vs T5: Finetuning for Classification

- **BART:** Repeat input in decoder



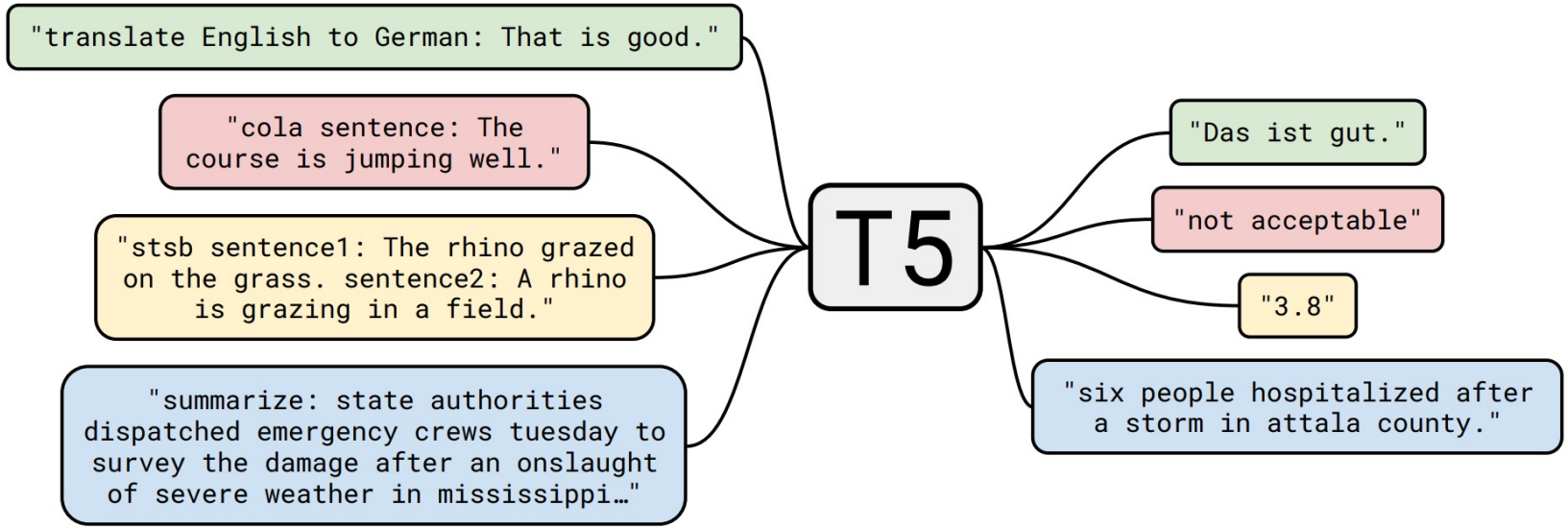
- **T5:** Repeat it as a seq2seq task



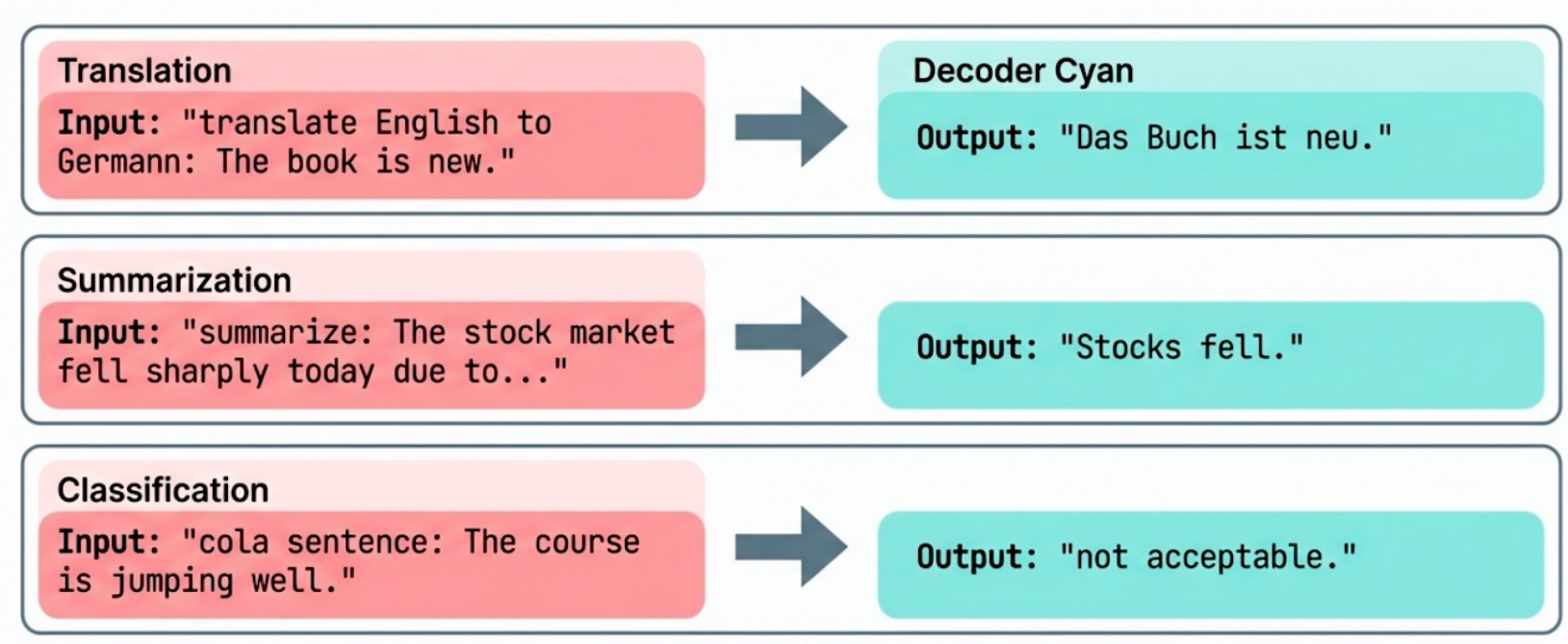
T5: Multi-Task Pre-Training Approach

The Power of Prefixes: One Model, Many Tasks

The T5 model leveraged multi-task pre-training on a diverse set of natural language tasks framed as text-to-text problems, enabling it to develop robust language understanding and generation capabilities that facilitated efficient transfer learning across various downstream tasks.

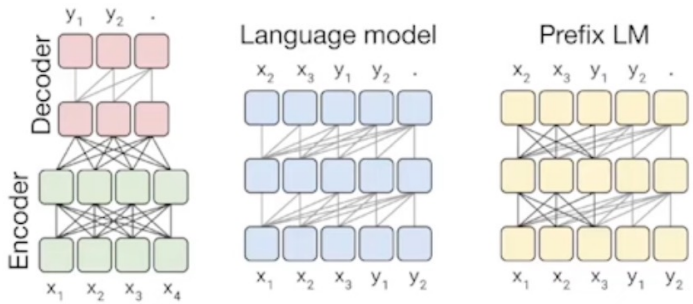


The Power of Prefixes: One Model, Many Tasks



No architectural changes required. The task is defined entirely by the input string.

Effectiveness of Denoising in T5



Architecture	Objective	Params	Cost	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Encoder-decoder	Denoising	$2P$	M	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
Enc-dec, shared	Denoising	P	M	82.81	18.78	**80.63**	**70.73**	26.72	39.03	**27.46**
Enc-dec, 6 layers	Denoising	P	$M/2$	80.88	18.97	77.59	68.42	26.38	38.40	26.95
Language model	Denoising	P	M	74.70	17.93	61.14	55.02	25.09	35.28	25.86
Prefix LM	Denoising	P	M	81.82	18.61	78.94	68.11	26.43	37.98	27.39
Encoder-decoder	LM	$2P$	M	79.56	18.59	76.02	64.29	26.27	39.17	26.86
Enc-dec, shared	LM	P	M	79.60	18.13	76.35	63.50	26.62	39.17	27.05
Enc-dec, 6 layers	LM	P	$M/2$	78.67	18.26	75.32	64.06	26.13	38.42	26.89
Language model	LM	P	M	73.78	17.54	53.81	56.51	25.23	34.31	25.38
Prefix LM	LM	P	M	79.68	17.84	76.87	64.86	26.28	37.51	26.76

BART vs T5: Model Architecture

- **Training data size**: BART > T5 (about 2x)
- **Model size**:
  - BART-large: 12 encoder, 12 decoder, 1024 hidden
  - T5-base: 12 encoder, 12 decoders, 768 hidden, 220M parameters (2x BERT-base)
  - T5-large: 24 encoder, 24 decoders, 1024 hidden, 770M parameters
- **Position encoding**:
  - BART: Learnable absolute position
  - T5: Relative position

BART vs T5: Performance

- **Language Understanding Performance**

	SQuAD	MNLI	SST	QQP	QNLI	STS-B	RTE	MRPC	CoLA
BART	88.8 / 94.6	89.9 / 90.1	96.6	92.5	94.9	91.2	87.2	90.4	62.8
T5	86.7 / 93.8	89.9 / 89.6	96.3	89.9	94.8	89.9	87.0	89.9	61.2

- **Generation Performance (Summarization)**

CNN/DailyMail	ROUGE-1	ROUGE-2	ROUGE-3
BART	45.14	21.28	37.25
T5	42.50	20.68	39.75

T5: Translation Example

```
from transformers import T5Tokenizer, T5ForConditionalGeneration


preprocessed_text = "translate English to German: the weather is good"
tokenizer = T5Tokenizer.from_pretrained('t5-base',
                                        max_length=64,
                                        model_max_length=512,
                                        legacy=False)
tokens = tokenizer.encode(preprocessed_text,
                          return_tensors="pt",
                          max_length=512,
                          truncation=True)

model = T5ForConditionalGeneration.from_pretrained('t5-base')
outputs = model.generate(tokens, min_length=4, max_length=32)

print("Result:", tokenizer.decode(outputs[0], skip_special_tokens=True))

#> Result: Das Wetter ist gut.
```

https://towardsdatascience.com/natural-language-processing-for-absolute-beginners-a195549a3164

T5: Summarization Example

```
body = '''Obviously, the lunar surface is covered with craters, left
from previous collisions of meteorites with the Moon. Where does math go?
While a meteorite collision is a random event, its frequency
obeys probability theory laws. There is no atmosphere on the Moon's
surface, no erosion, and no wind. Therefore the lunar surface is an
ideal "book" in which the events of the last tens of thousands of
years are recorded. By studying the Moon, we can calculate how often
such objects fall on its surface.

A study of the lunar surface with high-resolution cameras is ongoing.
It has been estimated that at least 220 new craters have formed on the
Moon over the past 7 years. This check is also vital because
these calculations can help assess the danger to the Earth.'''
```

```
preprocessed_text = f"summarize: {body}"

tokenizer = T5Tokenizer.from_pretrained('t5-base',
                                        max_length=256,
                                        model_max_length=512,
                                        legacy=False)
tokens = tokenizer.encode(preprocessed_text,
                          return_tensors="pt",
                          max_length=256,
                          truncation=True)

model = T5ForConditionalGeneration.from_pretrained('t5-base')
outputs = model.generate(tokens,
                         min_length=4,
                         max_length=64)

print("Result:", tokenizer.decode(outputs[0], skip_special_tokens=True))

#> the lunar surface is an ideal "book" in which the events of the last
#> tens of thousands of years are recorded. by studying the Moon, we can
#> calculate how often such objects are falling on its surface.
```

https://towardsdatascience.com/natural-language-processing-for-absolute-beginners-a195549a3164

Training: BERT vs BART vs T5 vs GPT

Pre-train => Fine-Tune (GPT, BERT, T5)

Domain Adaptation

- Typically requires many task-specific examples
- One specialized model for each task

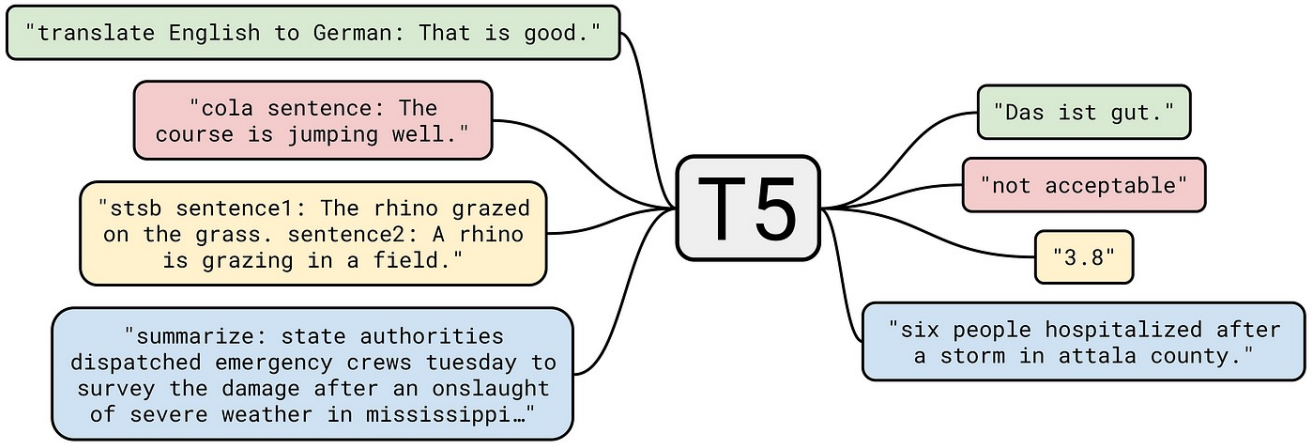
Pre-train => Prompting (GPT-2,GPT-3)

No Adaptation of the Model

- Just improve performance via few-shot prompting or prompt engineering

T5 Multitask Learning and Its Limitations

- Models like T5 trained on multiple tasks simultaneously.
  - T5's text-to-text format with task-specific prefixes improved performance on training tasks.
  - However, it failed to address cross-task generalization, making it unable to generalize to new, unseen tasks.



The T5 data format consists of a text-to-text format, where the input is a sequence of text with a task-specific prefix, and the output is a generated sequence of text.

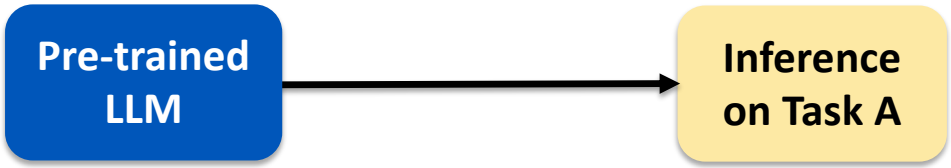
Domain Adaptation and Generalization Approaches

Pre-train => Fine-Tune (GPT, BERT, T5)

Domain Adaptation

- Typically requires many task-specific examples
- One specialized model for each task

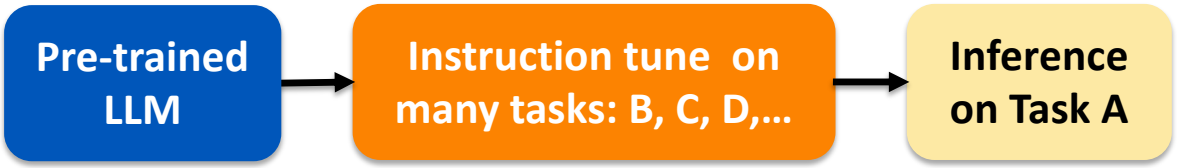
Pre-train => Prompting (GPT-2,GPT-3)



No Adaptation of the Model

- Just improve performance via few-shot prompting or prompt engineering

Pre-train => Instruction Tuning (FLAN)



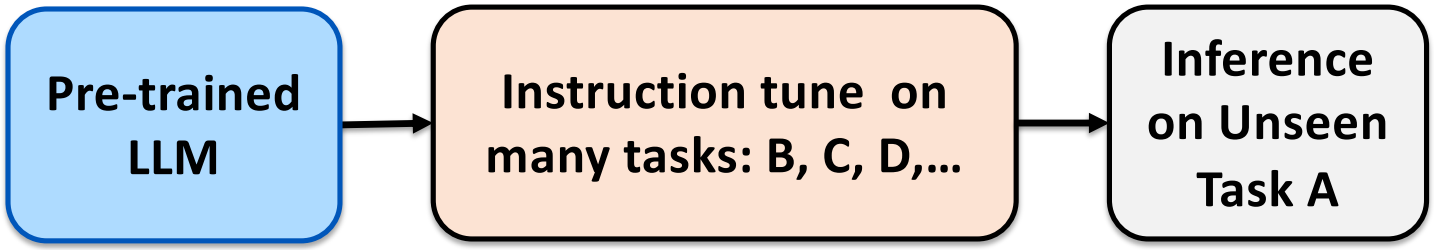
Domain Generalization

- **Model learns to perform many tasks via natural language instruction**
- Inference on unseen task

Instruction Tuning (Mid-2021)

- In 2021, **Instruction Tuning** emerged as a new approach to overcome traditional pre-training and fine-tuning limitations.
- **Instruction Tuning** involves **providing detailed, natural language instructions** during fine-tuning, enabling models to learn nuanced task context and relationships, and **generalize better to new tasks and contexts.**

Pre-train => Instruction Tuning (FLAN)



Instruction tuning is a process that refines LLMs to better understand and follow user instructions, enhancing their usefulness and responsiveness, and cross-task generalization capabilities.

Key Concepts of Instruction Tuning

- **Detailed Instructions:** Providing comprehensive, natural language descriptions of tasks **(Instruction with input context)** to help models understand and produce accurate responses. Example of Instruction-following data:

Instruction	**Input Context** (Optional)	**Response**
Translate English into Simplified Chinese	Welcome to Hong Kong	欢迎来到香港
Summarize in just 10 words to make the message even more brief and easier to remember.	The AAAI Conference on Artificial Intelligence, or AAAI, is a highly prestigious event organized by the Association for the Advancement of Artificial Intelligence. It gathers researchers, academics, and industry professionals globally to present and discuss the latest advancements, innovations, and applications in AI.	AAAI is a prestigious conference on artificial intelligence.

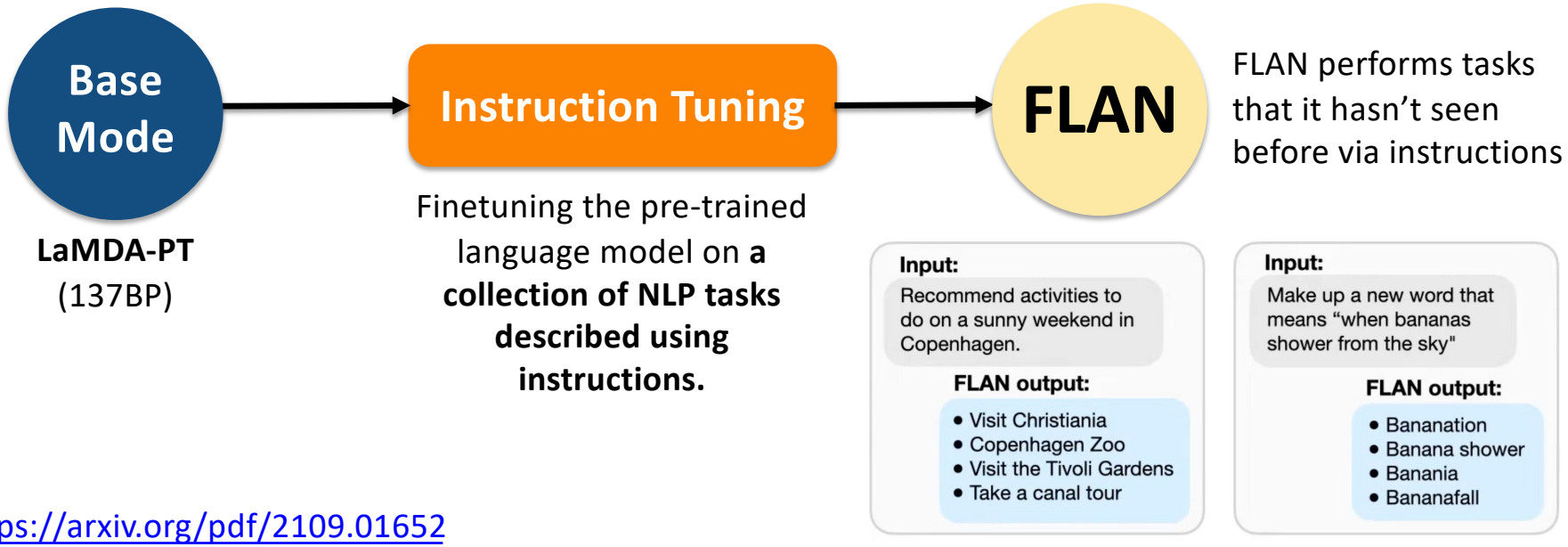
Key Concepts of Instruction Tuning

- **Generalization:** Training on various tasks with detailed instructions to enable models to generalize their understanding to new, unseen tasks.
- For example,

Instruction	**Input Context** (Optional)	**Response**
Summarize in just 10 words to make the message even more brief and easier to remember.	The AAAI Conference on Artificial Intelligence, or AAAI, is a highly prestigious event organized by the Association for the Advancement of Artificial Intelligence. It gathers researchers, academics, and industry professionals globally to present and discuss the latest advancements, innovations, and applications in AI.	AAAI 是人工智能领域最具声望和影响力的会议之一，由美国人工智能协会（AAAI）组织，每年举办一次，汇集全球顶尖的研究人员、学者和行业专家，分享最新的人工智能进展、创新和应用。

FLAN: Fine-tuned LAnguage Net (2021–09)

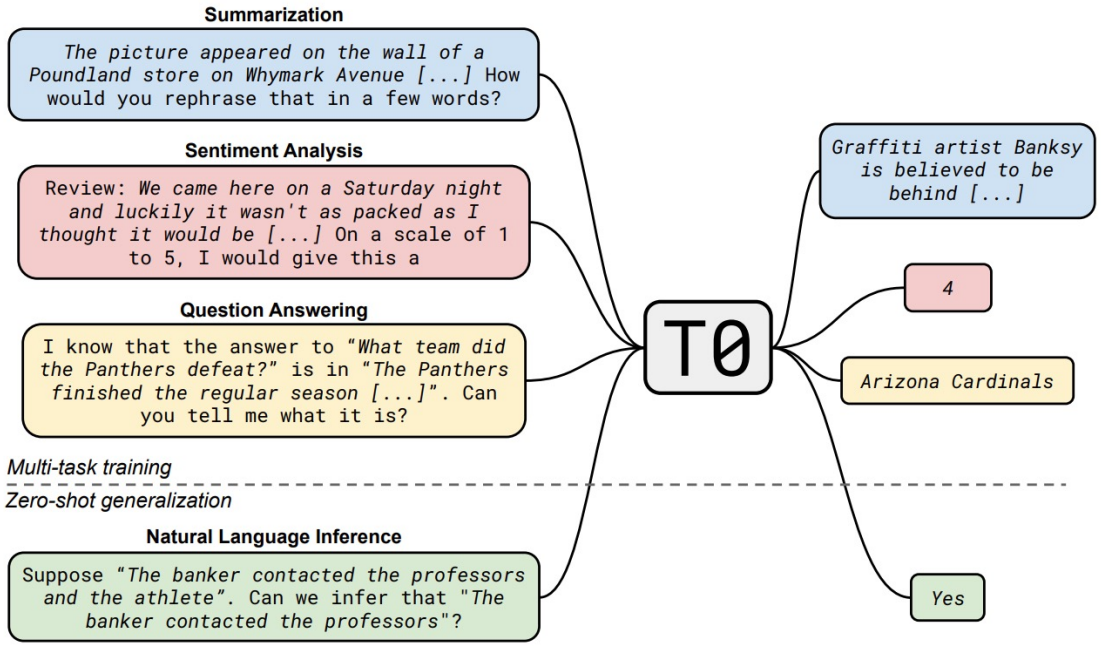
- Introduced by Wei et al. from Google, **FLAN** is a framework that fine-tunes language models with instructions, achieving significant performance improvements across various tasks.



https://arxiv.org/pdf/2109.01652

T0: Task-Specific Prompts (2021–10)

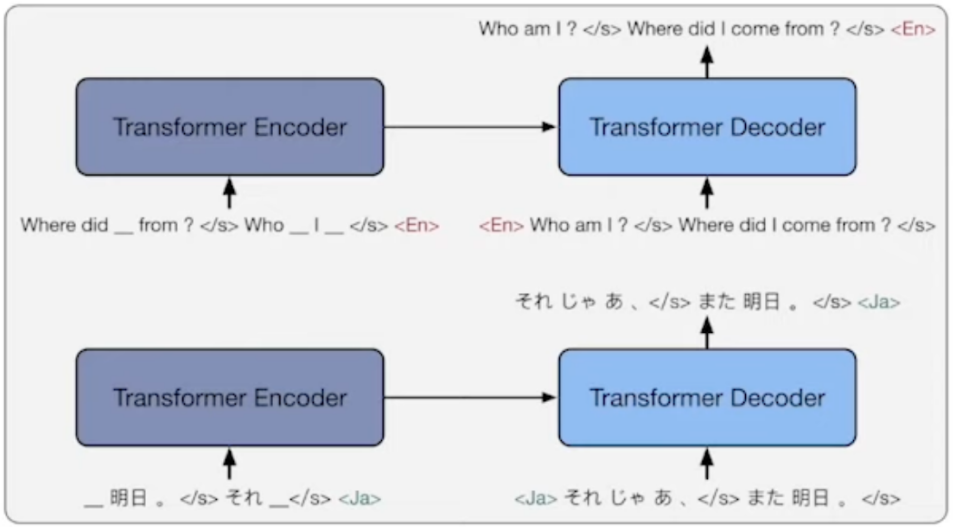
- T0 is a follow-up to the T5 model, which reframed NLP tasks as text-to-text problems. To overcome the T5 limitation, T0 was designed to enhance zero-shot learning through **task-specific prompts** during training.



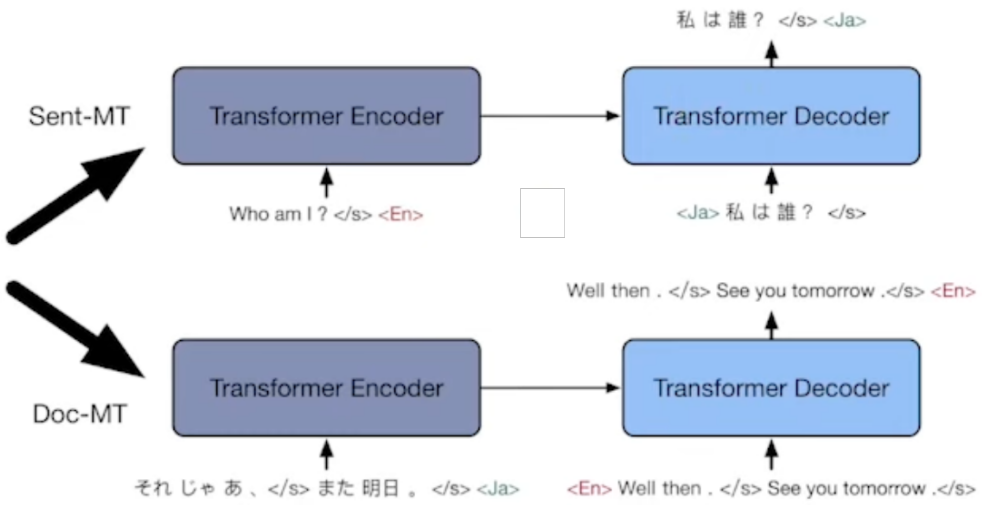
Other

Encoder-Decoder Transformer Models

mBART: Multilingual BART



Multilingual Denoising Pre-Training (mBART)



Fine-Tuning on Machine Translation

mT5: Multilingual T5 (2020-10)

Model	Architecture	Parameters	# languages	Data source
mBERT (Devlin, 2018)	Encoder-only	180M	104	Wikipedia
XLM (Conneau and Lample, 2019)	Encoder-only	570M	100	Wikipedia
XLM-R (Conneau et al., 2020)	Encoder-only	270M – 550M	100	Common Crawl (CCNet)
mBART (Lewis et al., 2020b)	Encoder-decoder	680M	25	Common Crawl (CC25)
MARGE (Lewis et al., 2020a)	Encoder-decoder	960M	26	Wikipedia or CC-News
mT5 (ours)	Encoder-decoder	300M – 13B	101	Common Crawl (mC4)

mT5

Model	Sentence pair		Structured	Question answering		
	XNLI	PAWS-X	WikiAnn NER	XQuAD	MLQA	TyDiQA-GoldP
Metrics	Acc.	Acc.	F1	F1 / EM	F1 / EM	F1 / EM
Cross-lingual zero-shot transfer (models fine-tuned on English data only)						
mBERT	65.4	81.9	62.2	64.5 / 49.4	61.4 / 44.2	59.7 / 43.9
XLM	69.1	80.9	61.2	59.8 / 44.3	48.5 / 32.6	43.6 / 29.1
InfoXLM	81.4	-	-	- / -	73.6 / 55.2	- / -
X-STILTs	80.4	87.7	64.7	77.2 / 61.3	72.3 / 53.5	76.0 / 59.5
XLM-R	79.2	86.4	65.4	76.6 / 60.8	71.6 / 53.2	65.1 / 45.0
VECO	79.9	88.7	65.7	77.3 / 61.8	71.7 / 53.2	67.6 / 49.1
RemBERT	80.8	87.5	**70.1**	79.6 / 64.0	73.1 / 55.0	77.0 / 63.0
mT5-Small	67.5	82.4	50.5	58.1 / 42.5	54.6 / 37.1	35.2 / 23.2
mT5-Base	75.4	86.4	55.7	67.0 / 49.0	64.6 / 45.0	57.2 / 41.2
mT5-Large	81.1	88.9	58.5	77.8 / 61.5	71.2 / 51.7	69.9 / 52.2
mT5-XL	82.9	89.6	65.5	79.5 / 63.6	73.5 / 54.5	75.9 / 59.4
mT5-XXL	**85.0**	**90.0**	69.2	**82.5 / 66.8**	**76.0 / 57.4**	**80.8 / 65.9**
Translate-train (models fine-tuned on English data plus translations in all target languages)						
XLM-R	82.6	90.4	-	80.2 / 65.9	72.8 / 54.3	66.5 / 47.7
FILTER + Self-Teaching	83.9	91.4	-	82.4 / 68.0	76.2 / 57.7	68.3 / 50.9
VECO	83.0	91.1	-	79.9 / 66.3	73.1 / 54.9	75.0 / 58.9
mT5-Small	64.7	79.9	-	64.3 / 49.5	56.6 / 38.8	48.2 / 34.0
mT5-Base	75.9	89.3	-	75.3 / 59.7	67.6 / 48.5	64.0 / 47.7
mT5-Large	81.8	91.2	-	81.2 / 65.9	73.9 / 55.2	71.1 / 54.9
mT5-XL	84.8	91.0	-	82.7 / 68.1	75.1 / 56.6	79.9 / 65.3
mT5-XXL	**87.8**	**91.5**	-	**85.2 / 71.3**	**76.9 / 58.3**	**82.8 / 68.8**
In-language multitask (models fine-tuned on gold data in all target languages)						
mBERT	-	-	89.1	-	-	77.6 / 68.0
mT5-Small	-	-	83.4	-	-	73.0 / 62.0
mT5-Base	-	-	85.4	-	-	80.8 / 70.0
mT5-Large	-	-	88.4	-	-	85.5 / 75.3
mT5-XL	-	-	90.9	-	-	87.5 / 78.1
mT5-XXL	-	-	**91.2**	-	-	**88.5 / 79.1**

M2M-100 (2021)

- The **M2M-100** model was proposed by Fan et al., (2021) in Beyond English-Centric Multilingual Machine Translation.
  - Before this model, the translations between two languages were **dependent on the English language.**
  - However, the M2M-100 model translates data from one language to another **without using English as an intermediary**.
  - This makes it **a true Many-to-Many multilingual translation model** that can translate directly between any pair of 100 languages.

https://dl.acm.org/doi/abs/10.5555/3546258.3546365

BigBird (2021)

- The **maximum context size** used in Transformer models is **limited** because it utilizes memory in terms of quadratic requirements.
- **BigBird** model solves this memory requirement challenge by using a sparse form of attention mechanism which enables it to scale in linear fashion.
- This allows for scaling from 512 tokens in most BERT models to **4096** tokens in BigBird
  - Heavily used in tasks like text summarization due to its ability to model long-term dependencies.