Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer: The T5 Model

Victoria Graf and Abhishek Panigrahi

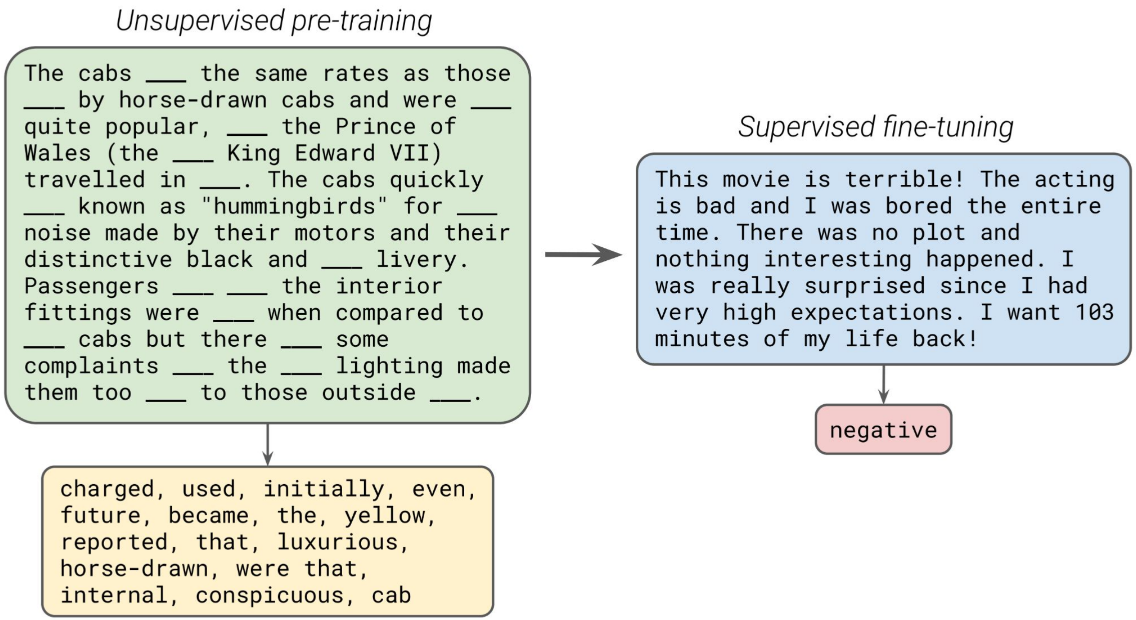
Transfer Learning

- Pre-training!
- Start with **unlabeled** data (unlike computer vision)
- General-purpose “English” knowledge

Transfer Learning

Transfer Learning

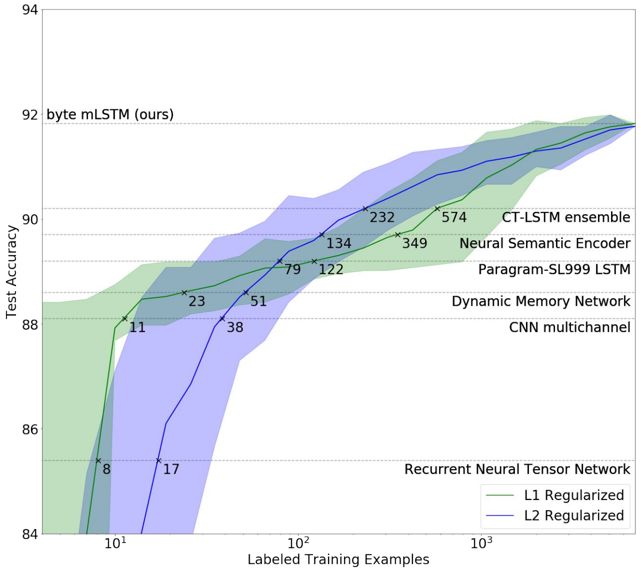
Transfer Learning



Slide adapted from Colin Raffel

A Very Brief Context

- 2017: Attention Is All You Need, Unsupervised sentiment neuron
- 2018: ELMo, GPT-1, BERT
  - Bidirectionality
  - Transformers
- 2019: RoBERTa, SpanBERT, ALBERT
- 2020: T5!



https://openai.com/blog/unsupervised-sentiment-neuron/

Transfer Learning: Comparisons?

Lots of research, so many…

- Pre-training objectives
- Unlabeled data sets
- Fine-tuning methods
- Model architectures/scales

… so how do we compare benchmarks?

Transfer Learning Comparisons

- Model A has 1B parameters and uses 100M pre-training tokens from <u>BooksCorpus</u>

Transfer Learning Comparisons

- Model A has 1B parameters and uses 100M pre-training tokens from <u>BooksCorpus</u>
- Model B is made by Google and their deep pockets! It has **2B** parameters and uses **200M** pre-training tokens from **<u>Wikipedia</u>**

Transfer Learning Comparisons

- Model A has 1B parameters and uses 100M pre-training tokens from BooksCorpus
- Model B is made by Google and their deep pockets! It has **2B** parameters and uses **200M** pre-training tokens from **Wikipedia**
- Model B has better performance on SuperGLUE than Model A

Transfer Learning Comparisons

- Model A has 1B parameters and uses 100M pre-training tokens from BooksCorpus
- Model B is made by Google and their deep pockets! It has **2B** parameters and uses **200M** pre-training tokens from **Wikipedia**
- Model B has better performance on SuperGLUE than Model A

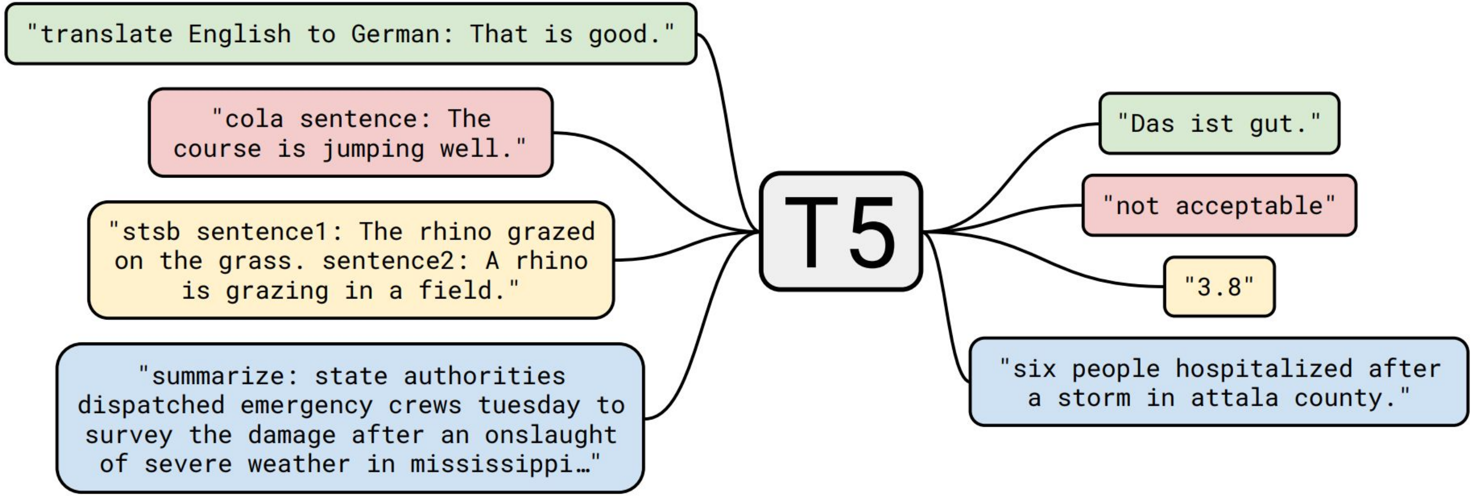
Is Wikipedia better for pre-training than BooksCorpus?

T5: The Basic Idea

- **T**ext-**t**o-**T**ext **T**ransfer **T**ransformer
- Every task, one format!
  - Previous attempts included:
    - Question answering
    - Language modeling
    - Span extraction

  … but had limitations
- "[Task-specific prefix]: [Input text]" -> "[output text]"

T5: The Basic Idea



T5: The Basic Idea

- GLUE and SuperGLUE **classification**; CNN/Daily Mail abstractive **summarization**; SQuAD **question answering**; and WMT English to German, French, and Romanian **translation**
  - GLUE/SuperGLUE: Sentence acceptability judgment, sentiment analysis, paraphrasing/sentence similarity, natural language inference, coreference resolution, sentence completion, word sense disambiguation, question answering
  - French: high resource, Romanian: low resource

- Separate fine-tuning for each task

Some tasks

Recall: SQuAD, GLUE benchmarks

- CoLA (GLUE): Sentence acceptability
  - **Input**: sentence, **output**: labels “acceptable” or “not acceptable”
  - Ex: “The course is jumping well.” -> not acceptable
- STS-B (GLUE): Sentence similarity
  - **Input**: pair of sentences, **output**: similarity score [1,5]
  - Ex: “sentence1: The rhino grazed. sentence2: A rhino is grazing.” -> 3.8

Some tasks

- COPA (SuperGLUE): Causal reasoning
  - **Input**: premise and 2 alternatives, **output**: alternative1 or alternative2
  - Ex: "Premise: I tipped the bottle. What happened as a RESULT?
    Alternative 1: The liquid in the bottle froze.
    Alternative 2: The liquid in the bottle poured out."
    -> alternative2
- ReCoRD/MultiRC (SuperGLUE): Question answering/Reading comprehension

Input/Output

[Task-specific prefix]: [Input text]

- EnDe (Translation):
  "translate English to German: That is good" -> "Das ist gut"

- CNNDM (Summarization):
  "summarize: state authorities dispatched…" -> "six people hospitalized after storm"

Input/Output

[Task-specific prefix]: [Input text]

- CoLA (GLUE; Classification):
  "cola sentence: The course is jumping well." -> "not acceptable"

- STS-B (GLUE; Regression):
  "stsb sentence1: The rhino grazed. sentence2: A rhino is grazing." -> "3.8"

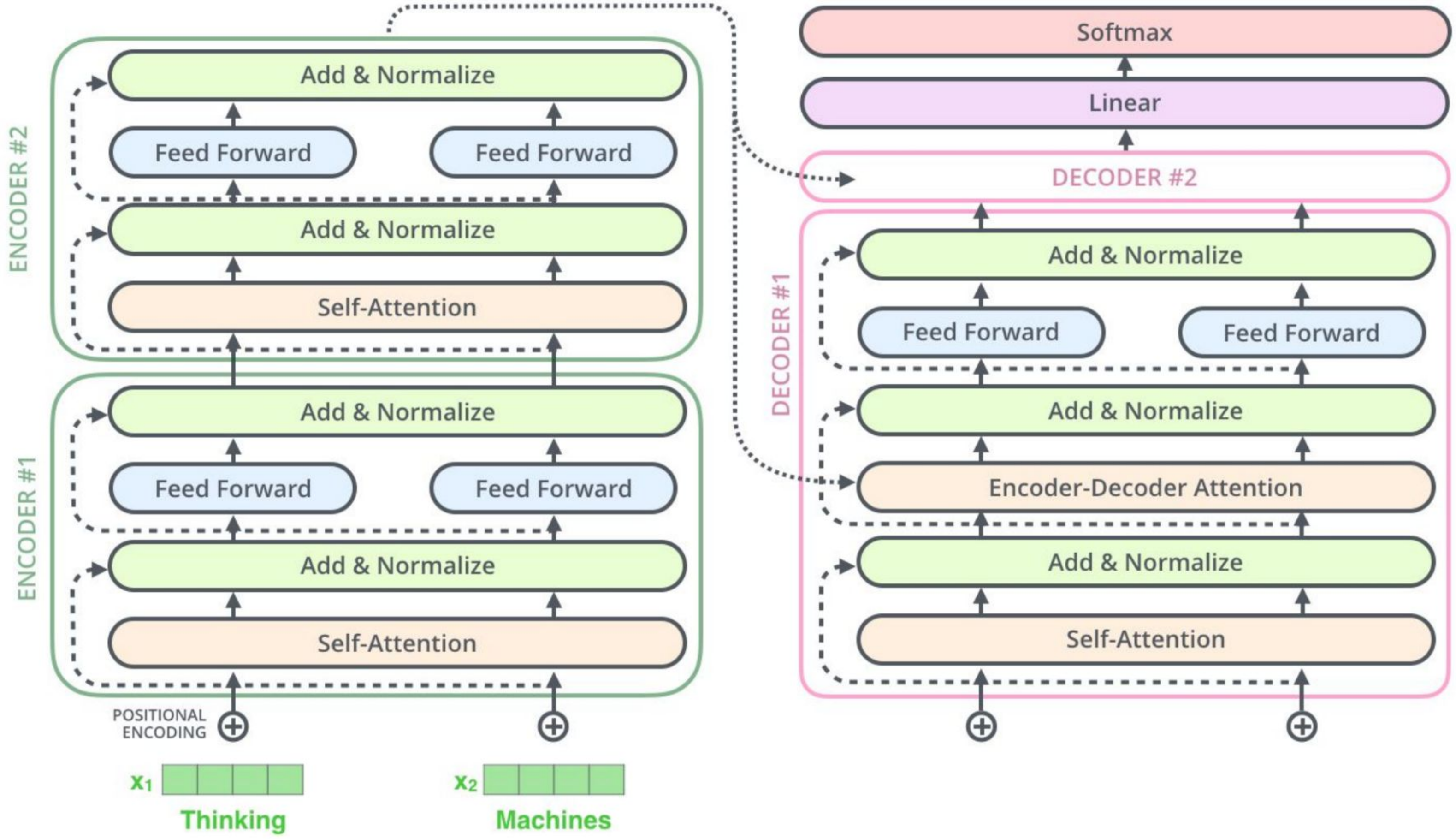
Input/Output

- CoLA (GLUE; Classification):
  “cola sentence: The course is jumping well.” -> “hamburger”

“Hamburger” is not a valid CoLA output, so this is a fail!

T5 Model

- Encoder-decoder model
  - Baseline size: two stacks of size $\text{BERT}_{\text{BASE}}$
- Architecture from “Attention Is All You Need”
  - Different position embedding scheme



http://jalammar.github.io/illustrated-transformer/

C4: The Data

- **C**olossal **C**lean **C**rawled **C**orpus
- Web-extracted text
- English language only (langdetect)
- 750GB

20TB to 750GB? Where did everything go?

C4: The Data

- Retain:
  - Sentences with terminal punctuation marks
  - Pages with at least 5 sentences, sentences with at least 3 words
- Deduplicate three sentence spans
- Remove:
  - References to Javascript
  - Lorem ipsum text
  - Code

C4: The Data

Menu

Lemon

Introduction

The lemon, Citrus Limon (l.) Osbeck, is a species of small evergreen tree in the flowering plant family rutaceae.
The tree's ellipsoidal yellow fruit is used for culinary and non-culinary purposes throughout the world, primarily for its juice, which has both culinary and cleaning uses.
The juice of the lemon is about 5% to 6% citric acid, with a ph of around 2.2, giving it a sour taste.

Article

The origin of the lemon is unknown, though lemons are thought to have first grown in Assam (a region in northeast India), northern Burma or China.
A genomic study of the lemon indicated it was a hybrid between bitter orange (sour orange) and citron.

Please enable JavaScript to use our site.

Home
Products
Shipping
Contact
FAQ

Dried Lemons, $3.59/pound

Organic dried lemons from our farm in California.
Lemons are harvested and sun-dried for maximum flavor.
Good in soups and on popcorn.

The lemon, Citrus Limon (l.) Osbeck, is a species of small evergreen tree in the flowering plant family rutaceae.
The tree's ellipsoidal yellow fruit is used for culinary and non-culinary purposes throughout the world, primarily for its juice, which has both culinary and cleaning uses.
The juice of the lemon is about 5% to 6% citric acid, with a ph of around 2.2, giving it a sour taste.

Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Curabitur in tempus quam. In mollis et ante at consectetur.
Aliquam erat volutpat.
Donec at lacinia est.
Duis semper, magna tempor interdum suscipit, ante elit molestie urna, eget efficitur risus nunc ac elit.
Fusce quis blandit lectus.
Mauris at mauris a turpis tristique lacinia at nec ante.
Aenean in scelerisque tellus, a efficitur ipsum.
Integer justo enim, ornare vitae sem non, mollis fermentum lectus.
Mauris ultrices nisl at libero porta sodales in ac orci.

```
function Ball(r) {
  this.radius = r;
  this.area = pi * r ** 2;
  this.show = function(){
    drawCircle(r);
  }
}
```

Slide adapted from Colin Raffel

C4: The Data

Menu

Lemon

Introduction

The lemon, Citrus Limon (l.) Osbeck, is a species of small evergreen tree in the flowering plant family rutaceae.
The tree's ellipsoidal yellow fruit is used for culinary and non-culinary purposes throughout the world, primarily for its juice, which has both culinary and cleaning uses.
The juice of the lemon is about 5% to 6% citric acid, with a ph of around 2.2, giving it a sour taste.

Article

The origin of the lemon is unknown, though lemons are thought to have first grown in Assam (a region in northeast India), northern Burma or China.
A genomic study of the lemon indicated it was a hybrid between bitter orange (sour orange) and citron.

Please enable JavaScript to use our site.

Home
Products
Shipping
Contact
FAQ

Dried Lemons, $3.59/pound

Organic dried lemons from our farm in California.
Lemons are harvested and sun-dried for maximum flavor.
Good in soups and on popcorn.

The lemon, Citrus Limon (l.) Osbeck, is a species of small evergreen tree in the flowering plant family rutaceae.
The tree's ellipsoidal yellow fruit is used for culinary and non-culinary purposes throughout the world, primarily for its juice, which has both culinary and cleaning uses.
The juice of the lemon is about 5% to 6% citric acid, with a ph of around 2.2, giving it a sour taste.

Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Curabitur in tempus quam. In mollis et ante at consectetur.
Aliquam erat volutpat.
Donec at lacinia est.
Duis semper, magna tempor interdum suscipit, ante elit molestie urna, eget efficitur risus nunc ac elit.
Fusce quis blandit lectus.
Mauris at mauris a turpis tristique lacinia at nec ante.
Aenean in scelerisque tellus, a efficitur ipsum.
Integer justo enim, ornare vitae sem non, mollis fermentum lectus.
Mauris ultrices nisl at libero porta sodales in ac orci.

```
function Ball(r) {
  this.radius = r;
  this.area = pi * r ** 2;
  this.show = function(){
    drawCircle(r);
  }
}
```

Slide adapted from Colin Raffel

C4: The Data

750GB? What does that mean?

	Data set	Size
★	C4	745GB
	C4, unfiltered	6.1TB
	RealNews-like	35GB
	WebText-like	17GB
	Wikipedia	16GB
	Wikipedia + TBC	20GB

Vocabulary

- 32,000 wordpieces shared across input and output
- Pre-training is English, but fine-tuning includes German, French, and Romanian
- Trained SentencePiece model 10:1:1:1 English : German : French : Romanian
    - Can handle fixed set of languages

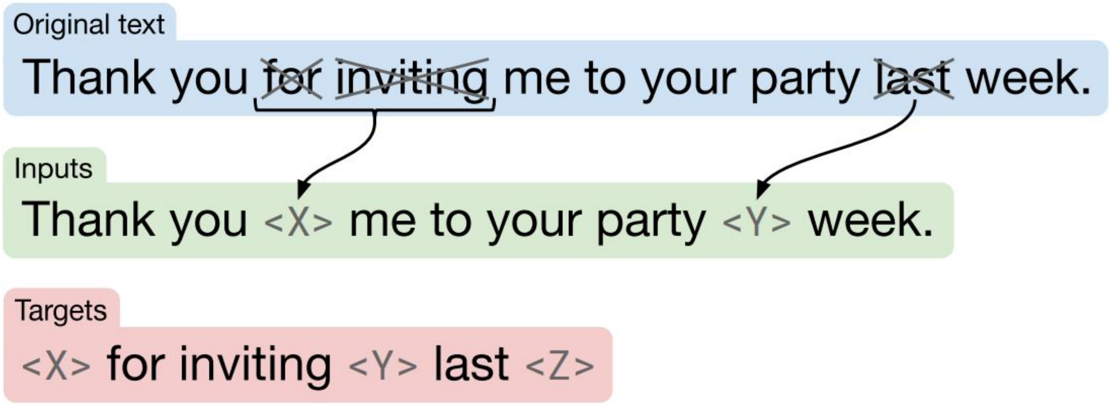
mT5

- mC4: Common Crawl dataset covering 101 languages!
  - Only line length filter, no punctuation filter
  - How do you sample across languages?
    - “Boosting” the probability of training on low-resource languages without overfitting
- Similar architecture to T5
- 6 tasks from the XTREME multilingual benchmark
  - Entailment, reading comprehension, NER, paraphrase identification
- Illegal predictions (XQuAD)

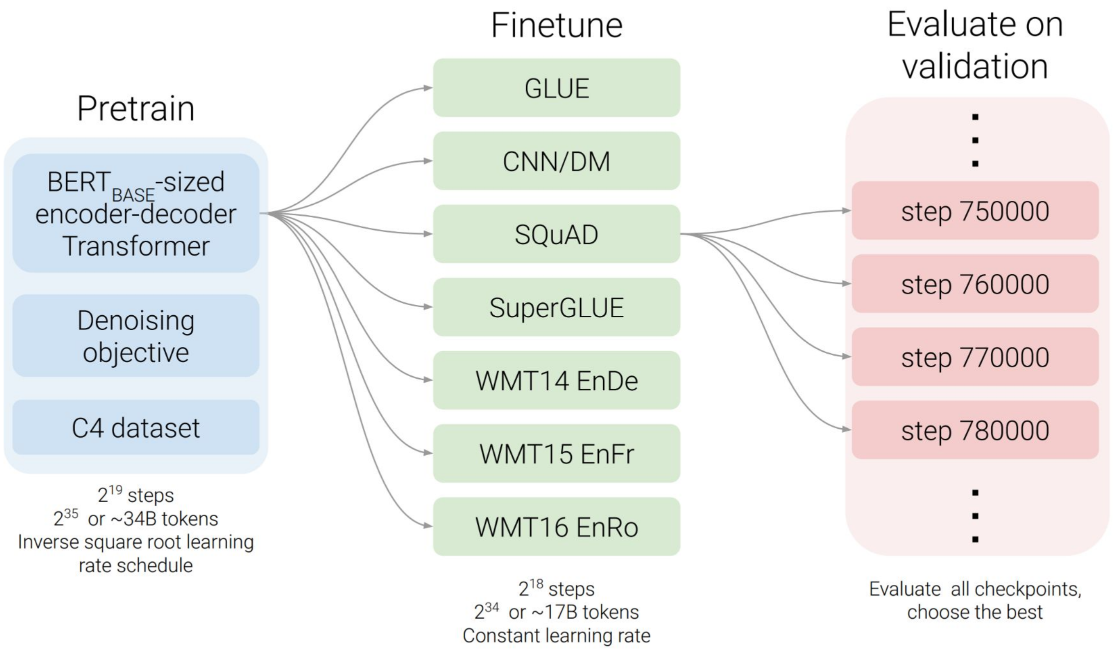
Experiments

- Even Google has a budget…
  - NOT combinatorial
  - Standard deviation only found for baseline
  - ~2^{35} or 34B pre-training tokens (much less than BERT!)
- Inverse square-root learning rate schedule with warm-up
- Results reported on validation sets

Baseline Objective



Workflow



Slide adapted from Colin Raffel

Baseline Performance

Bold scores are within two standard deviations of the best score in a given experiment

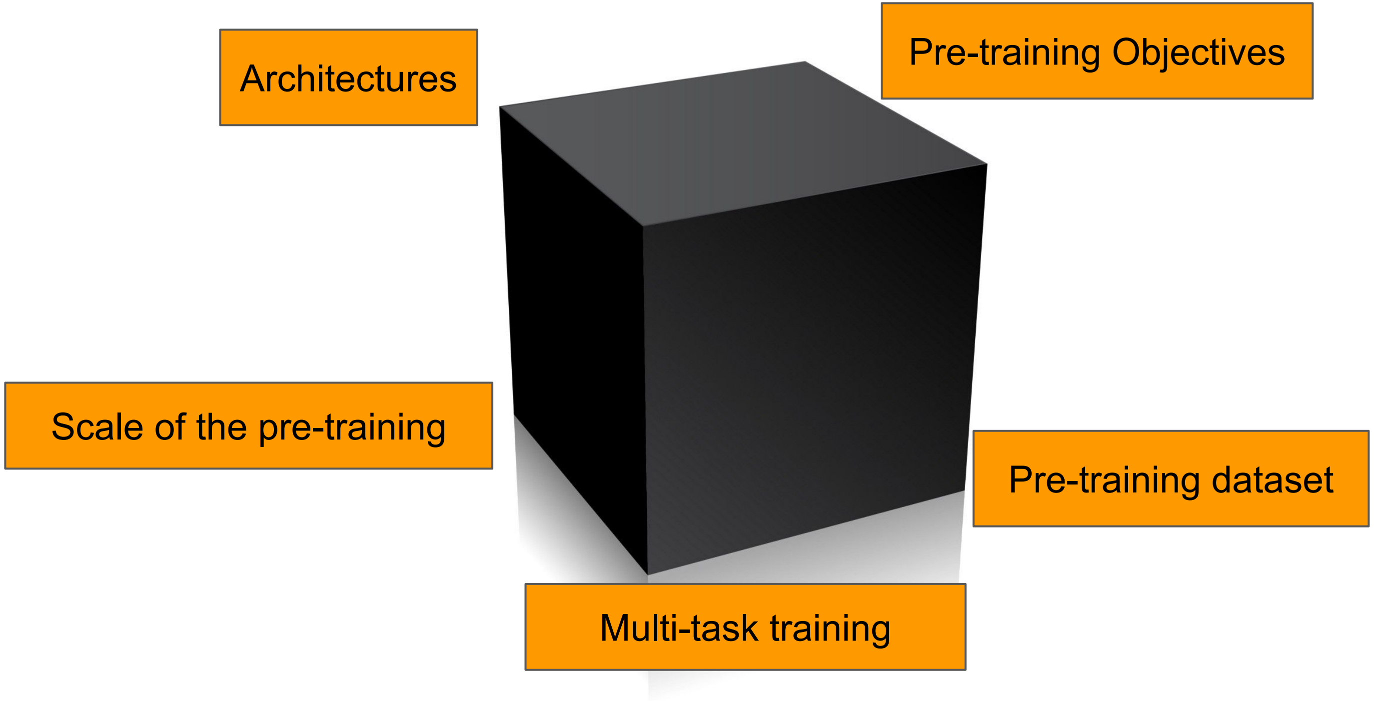
	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Baseline average	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
Baseline standard deviation	0.235	0.065	0.343	0.416	0.112	0.090	0.108
No pre-training	66.22	17.60	50.31	53.04	25.86	**39.77**	24.04

Baseline Performance

- GLUE/SuperGLUE are sets of tasks including CoLA, STS-B, etc.
- CNNDM is a summarization task
- EnDe/EnFr/EnRo are translation tasks

	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Baseline average	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
Baseline standard deviation	0.235	0.065	0.343	0.416	0.112	0.090	0.108
No pre-training	66.22	17.60	50.31	53.04	25.86	**39.77**	24.04

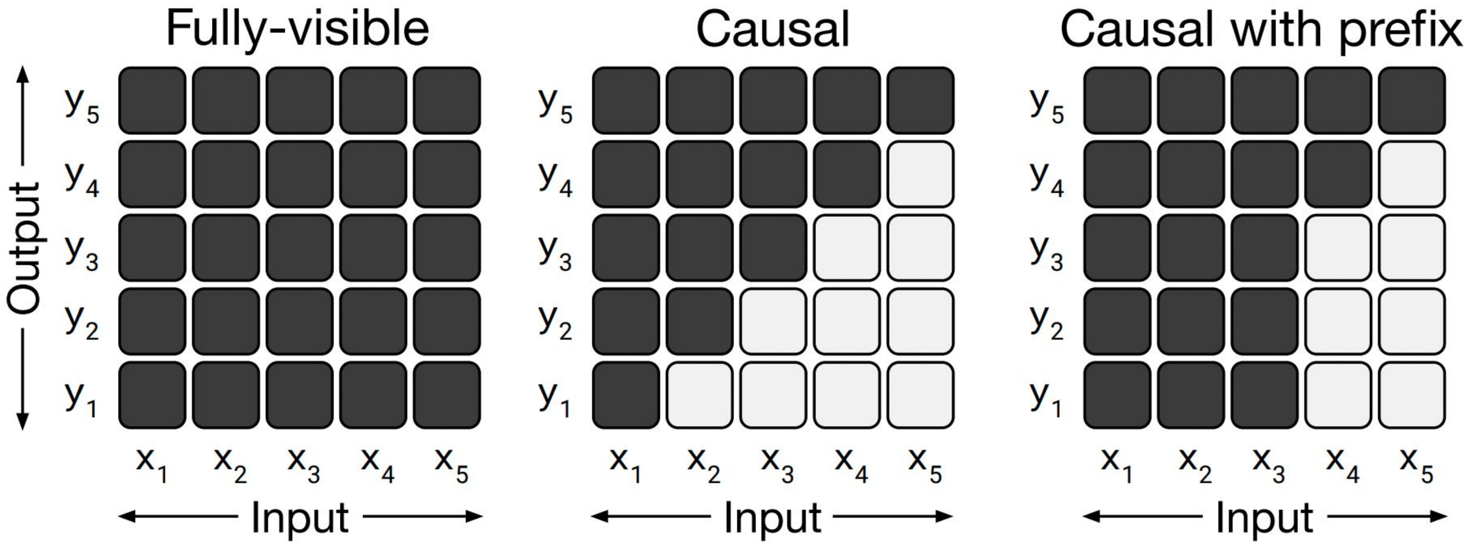
Axis of Decisions for Pre-training and Fine-tuning



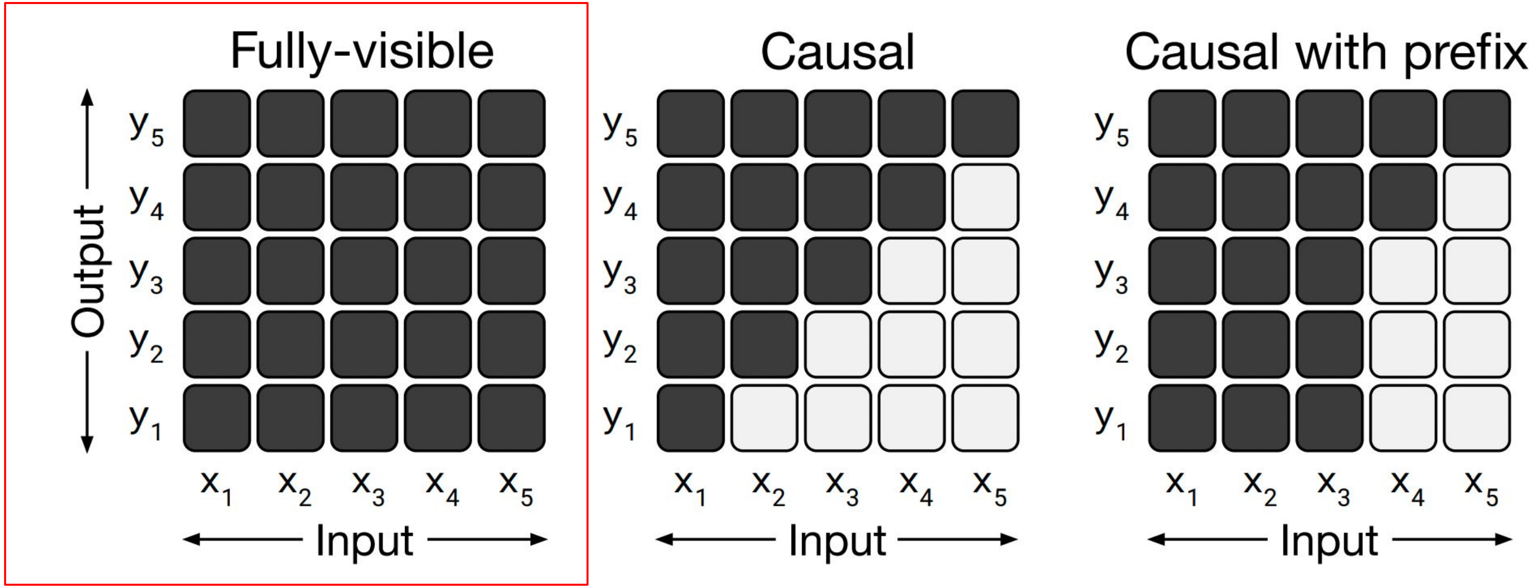
Understand the first order effect of each aspect by altering it while keeping other aspects of pre-training fixed.

Architectures

Different Attention Mask Patterns

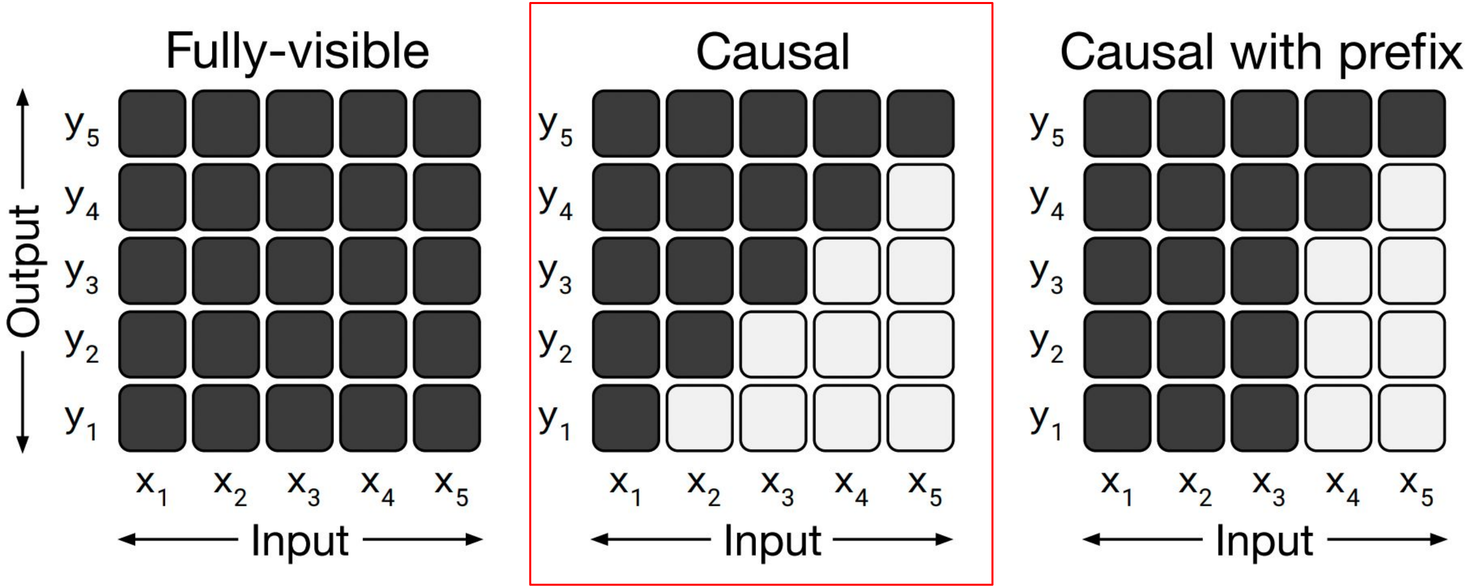


Different Attention Mask Patterns



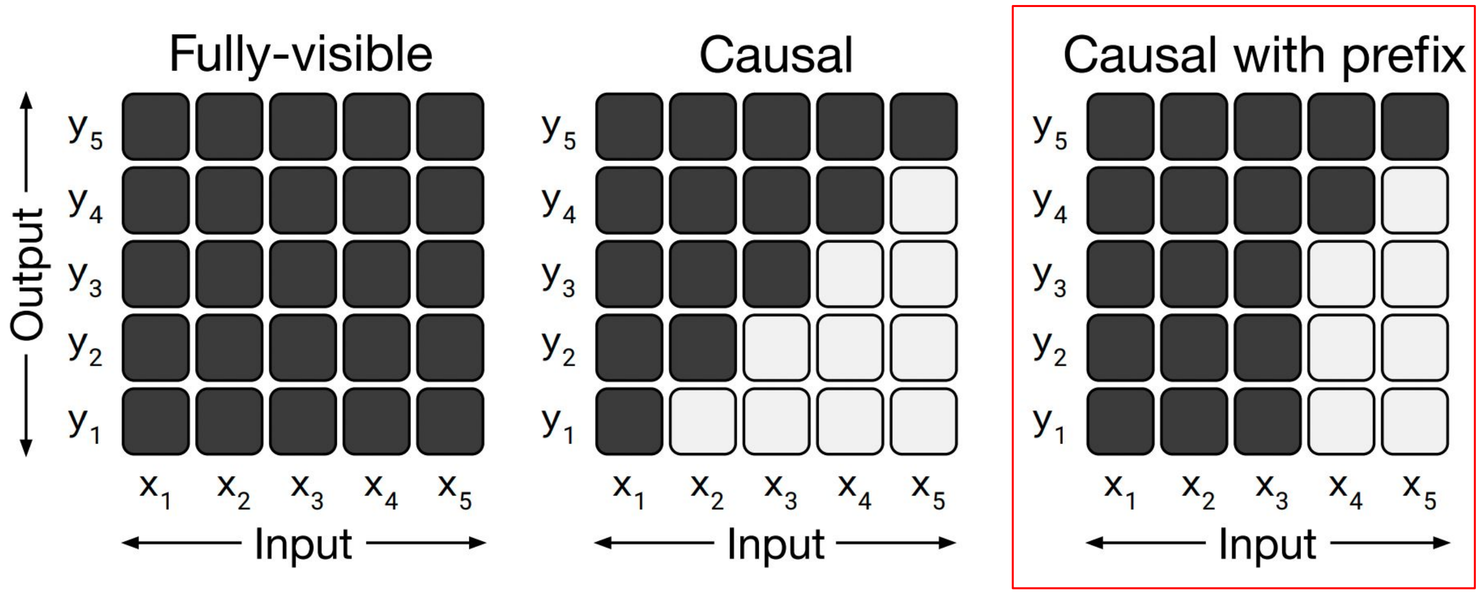
Fully visible mask allows the self attention mechanism to attend to the full input.

Different Attention Mask Patterns



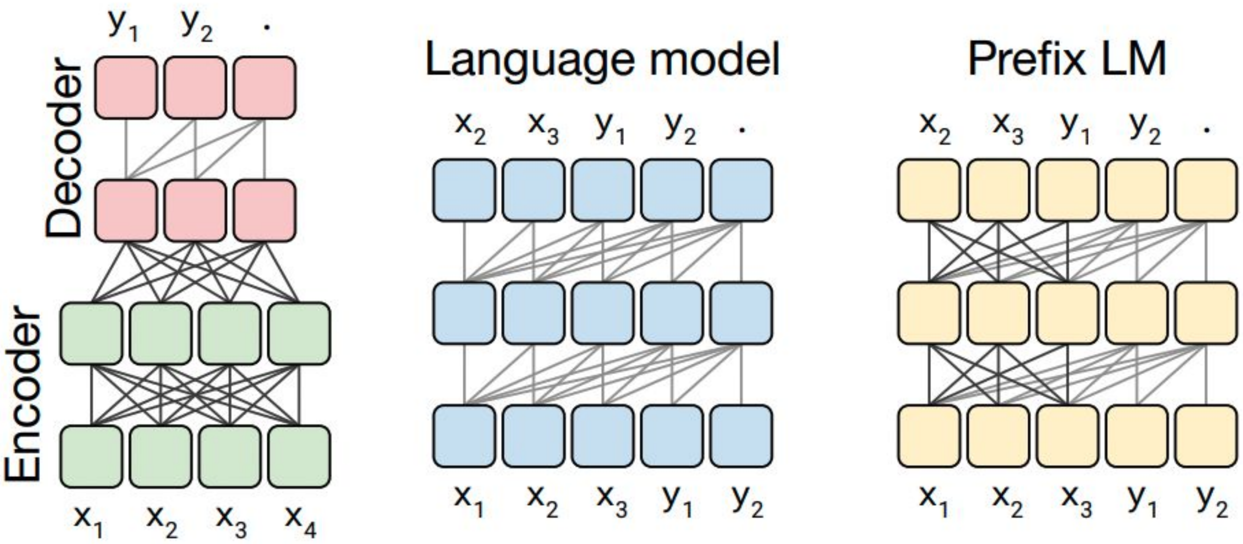
A causal mask doesn't allow output elements to look into the future.

Different Attention Mask Patterns

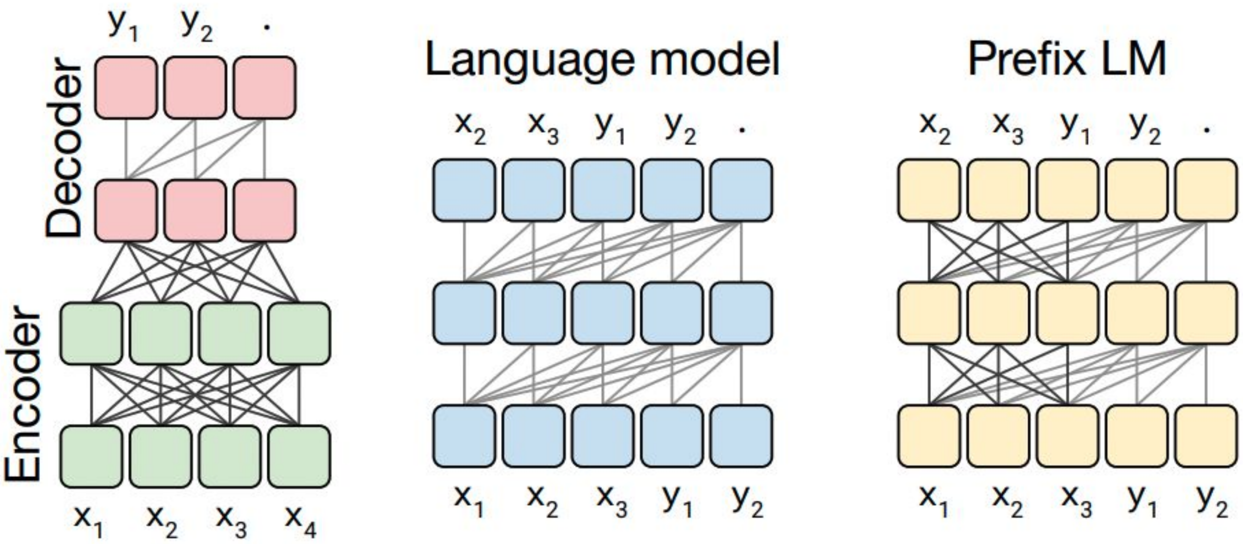


Causal mask with prefix allows to fully-visible masking on a portion of input.

Transformer Architecture Variants

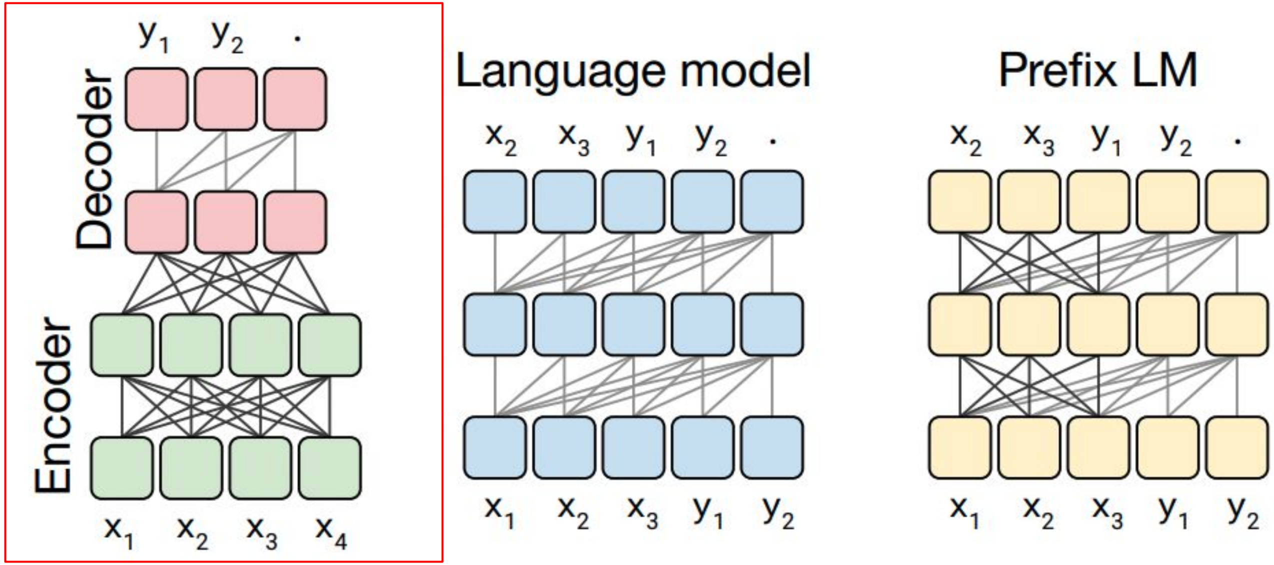


Transformer Architecture Variants



Translation: That is good -> Das ist gut.

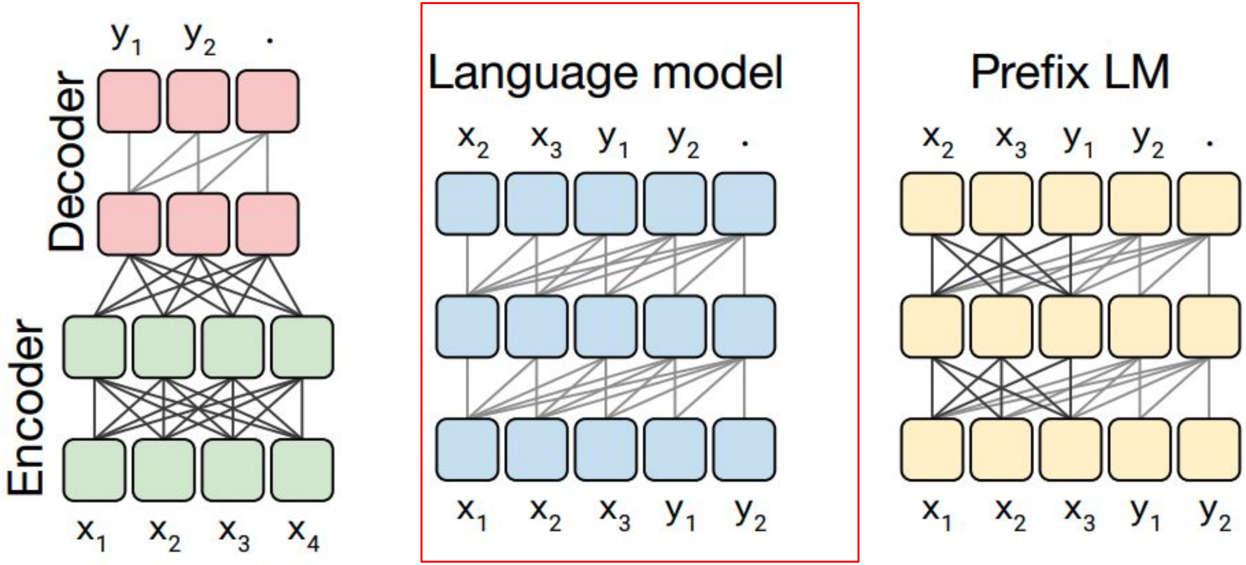
Transformer Architecture Variants



Translation: That is good -> Das ist gut.

Translate English to German: That is good. Target: Das is gut.

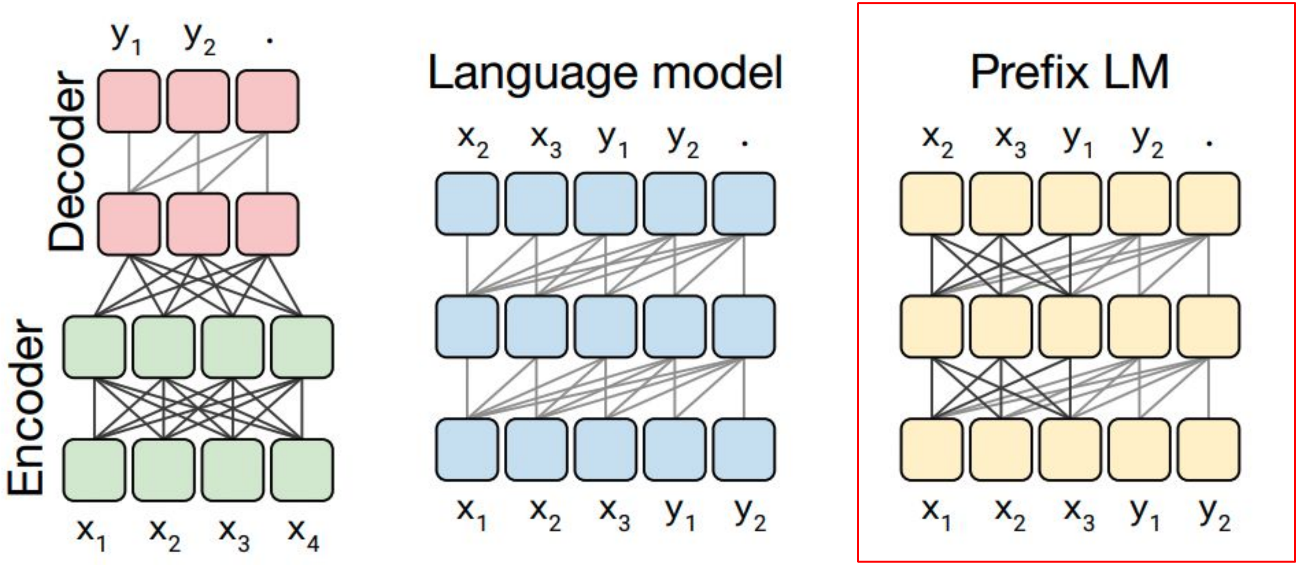
Transformer Architecture Variants



Translation: That is good -> Das ist gut.

- Translate English to German: That is good. Target: Das is gut.
    - “Good” representation can only look at “Translate English to German: That is”.

Transformer Architecture Variants

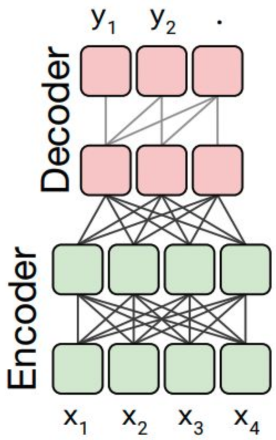


Translation: That is good -> Das ist gut.

- Translate English to German: That is good. Target: Das is gut.
  - "Good" representation can look at "Translate English to German: That is. Target:".

Performance of different Architectural Variants

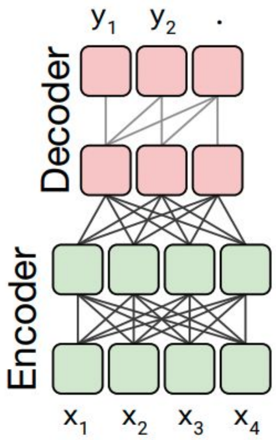
Architecture	Objective	Params	Cost	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Encoder-decoder	Denoising	$2P$	M	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**



Performance of different Architectural Variants

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**

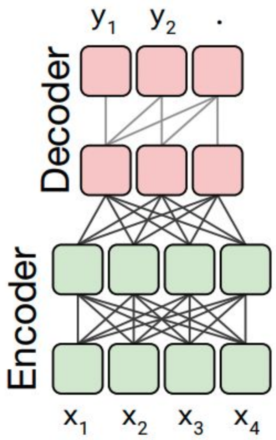
Input: Thank you for <X> me to your party <Y>.
Target: <X> inviting <Y> last week.



Performance of different Architectural Variants

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**

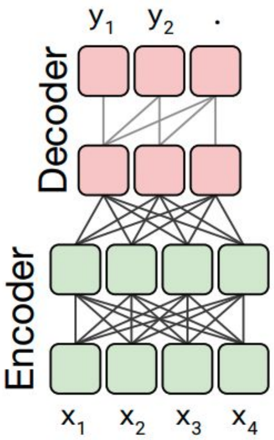
Number of parameters



Performance of different Architectural Variants

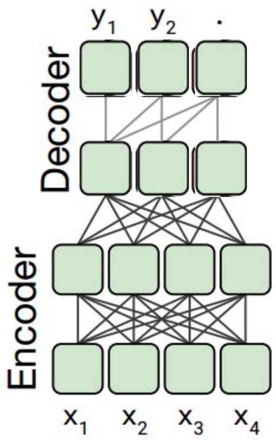
Architecture	Objective	Params	Cost	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Encoder-decoder	Denoising	$2P$	M	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**

Number of flops



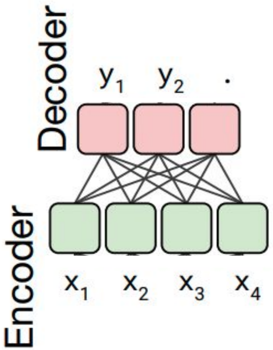
Performance of different Architectural Variants

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**



Performance of different Architectural Variants

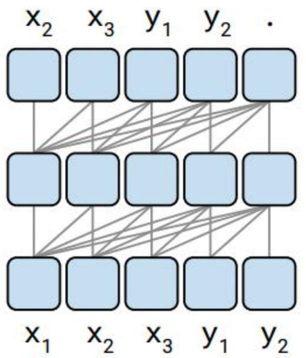
Architecture	Objective	Params	Cost	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Encoder-decoder	Denoising	$2P$	M	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
Enc-dec, shared	Denoising	P	M	82.81	18.78	**80.63**	**70.73**	26.72	39.03	**27.46**
Enc-dec, 6 layers	Denoising	P	$M/2$	80.88	18.97	77.59	68.42	26.38	38.40	26.95



Performance of different Architectural Variants

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

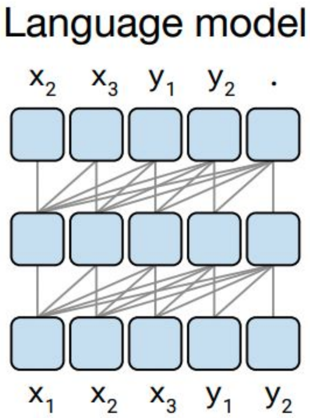
Language model



Performance of different Architectural Variants

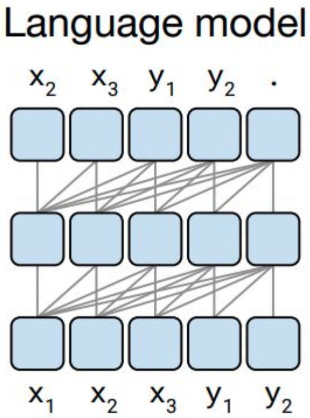
	Architecture	Objective	Params	Cost	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★	Encoder-decoder	Denoising	$2P$	M	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
	Enc-dec, shared	Denoising	P	M	82.81	18.78	**80.63**	**70.73**	26.72	39.03	**27.46**
	Enc-dec, 6 layers	Denoising	P	$M/2$	80.88	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

Language model is decoder-only



Performance of different Architectural Variants

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

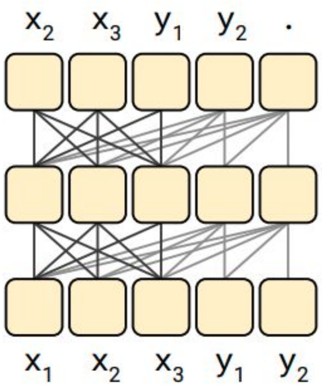


LM looks at both input and target, while encoder only looks at input sequence and decoder looks at output sequence.

Performance of different Architectural Variants

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

Prefix LM



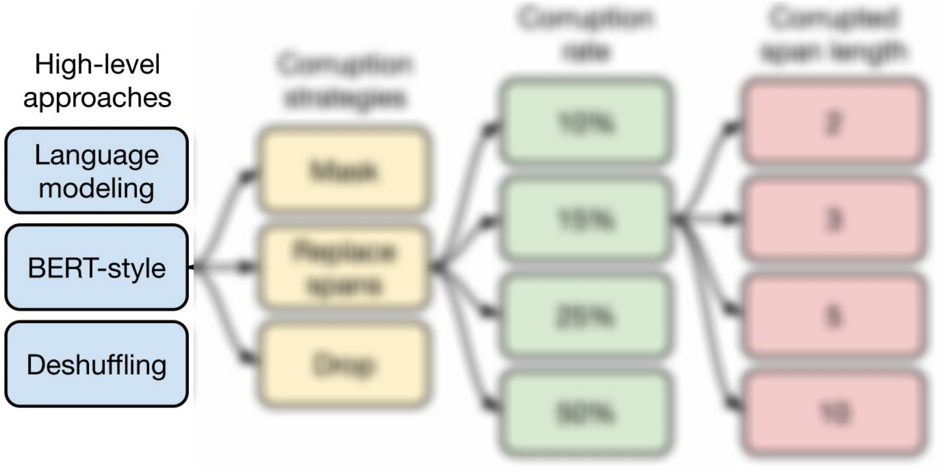
Performance of different Architectural Variants

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

1. Sharing parameters in encoder and decoder models perform nearly as well as the baseline.
2. Halving the number of layers in encoder and decoder hurts the performance.
3. Performance of Encoder and Decoder with shared parameters is better than decoder only LM and prefix LM.

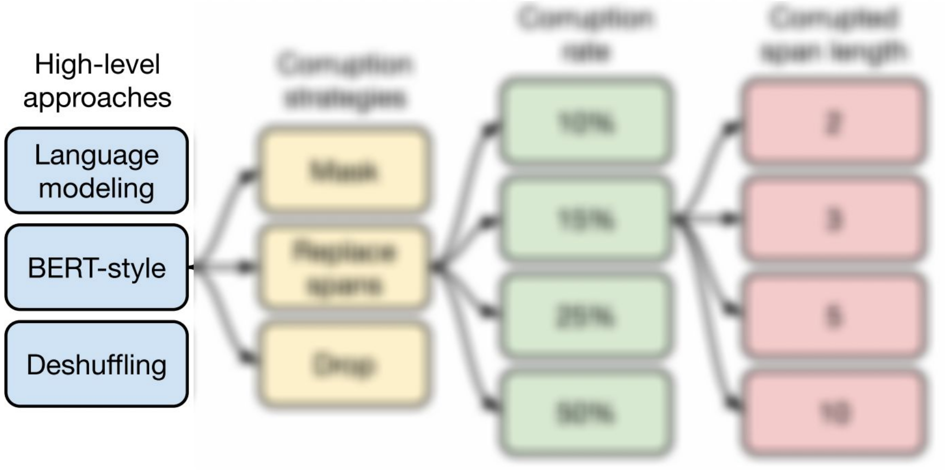
Objectives

Different Unsupervised Objectives



Objective	Inputs	Targets
Prefix language modeling	Thank you for inviting	me to your party last week .
BERT-style Devlin et al. (2018)	Thank you <M> <M> me to your party apple week .	*(original text)*
Deshuffling	party me for your to . last fun you inviting week Thank	*(original text)*

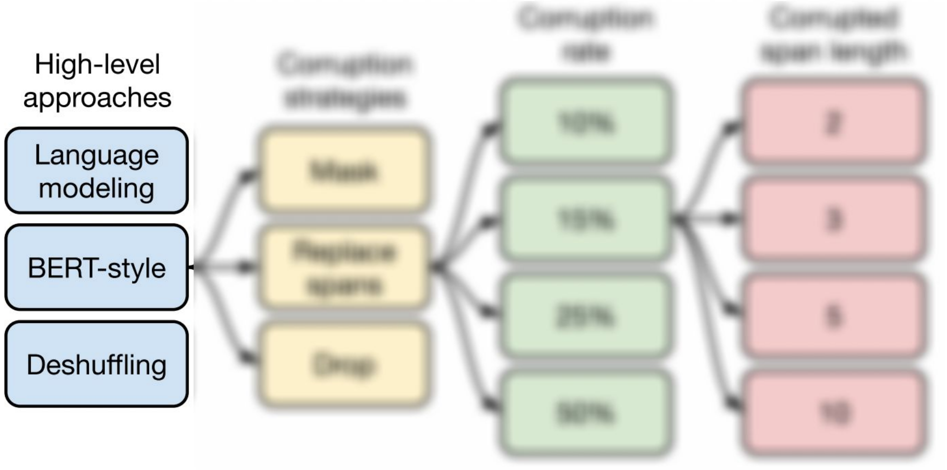
Different Unsupervised Objectives



Objective	Inputs	Targets
Prefix language modeling	Thank you for inviting	me to your party last week .
BERT-style Devlin et al. (2018)	Thank you <M> <M> me to your party apple week .	*(original text)*
Deshuffling	party me for your to . last fun you inviting week Thank	*(original text)*

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

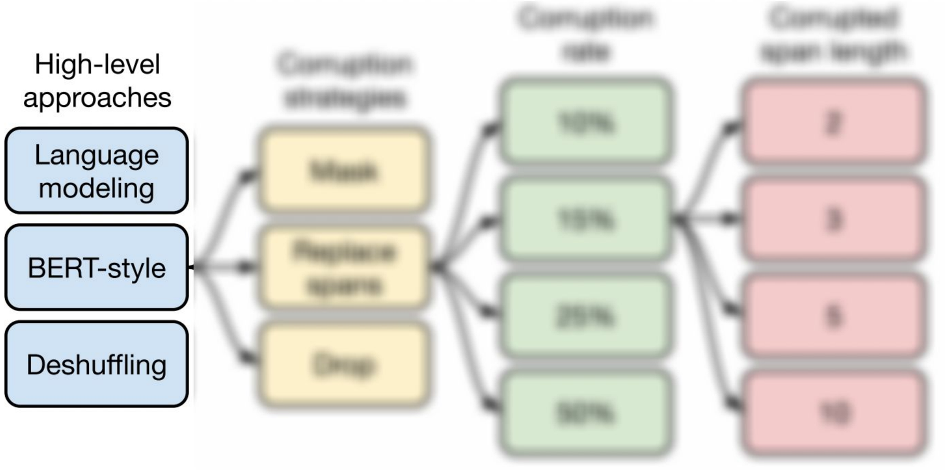
Different Unsupervised Objectives



Objective	Inputs	Targets
Prefix language modeling	Thank you for inviting	me to your party last week .
BERT-style Devlin et al. (2018)	Thank you <M> <M> me to your party apple week .	*(original text)*
Deshuffling	party me for your to . last fun you inviting week Thank	*(original text)*

- Thank you <M> <M> me to your party apple week . Thank you for inviting me to your party last week.

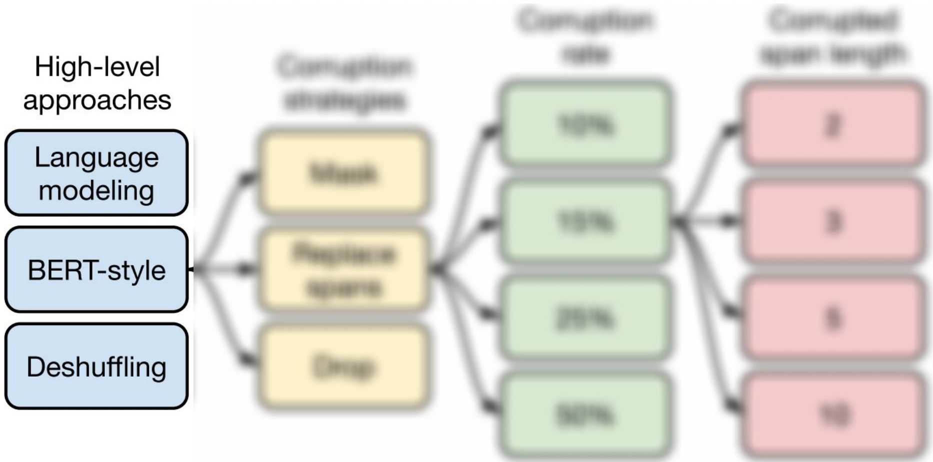
Different Unsupervised Objectives



Objective	Inputs	Targets
Prefix language modeling	Thank you for inviting	me to your party last week .
BERT-style Devlin et al. (2018)	Thank you <M> <M> me to your party apple week .	*(original text)*
Deshuffling	party me for your to . last fun you inviting week Thank	*(original text)*

- party me for your to . last fun you inviting week Thank . Thank you for inviting me to your party last week.

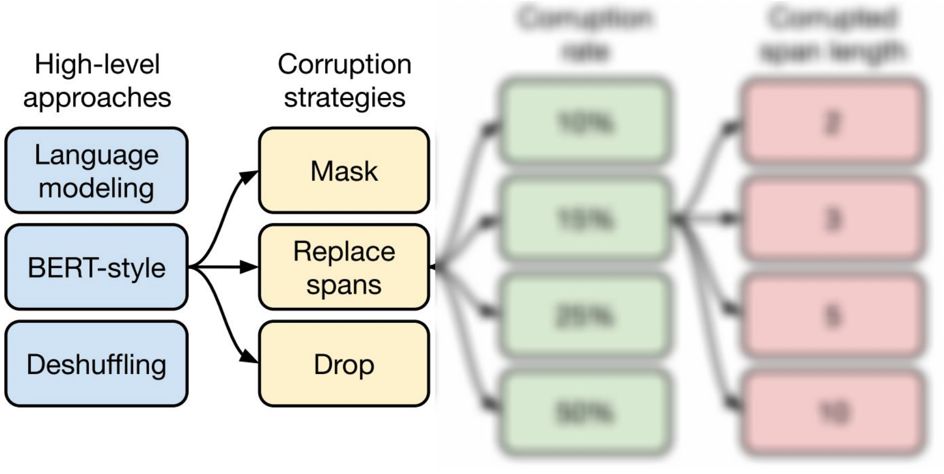
Performance of the three disparate pre-training objectives



1. BERT-style objective performs best.
2. Prefix LM works well on translation tasks.
3. Deshuffling objective is significantly worse.

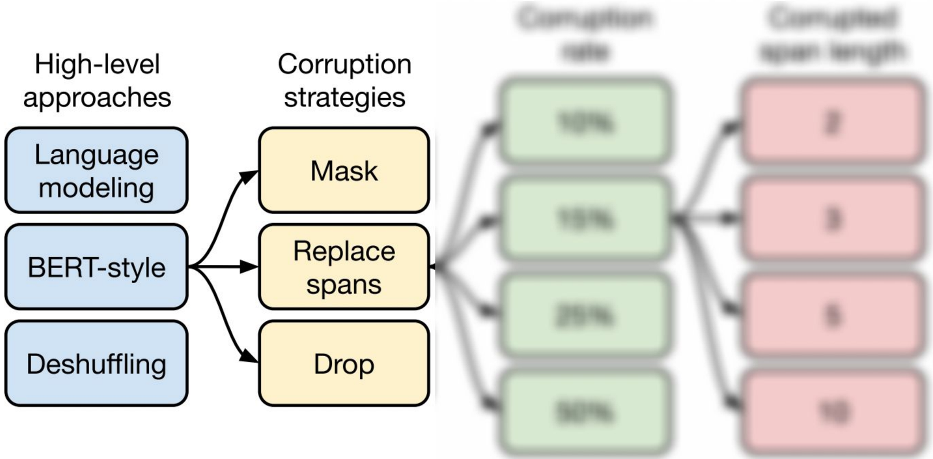
Objective	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
Prefix language modeling	80.69	18.94	77.99	65.27	**26.86**	39.73	**27.49**
BERT-style (Devlin et al., 2018)	**82.96**	**19.17**	**80.65**	**69.85**	**26.78**	**40.03**	**27.41**
Deshuffling	73.17	18.59	67.61	58.47	26.11	39.30	25.62

Different BERT-style Unsupervised Objectives



Objective	Inputs	Targets
MASS-style Song et al. (2019)	Thank you <M> <M> me to your party <M> week .	*(original text)*
I.i.d. noise, replace spans	Thank you <X> me to your party <Y> week .	<X> for inviting <Y> last <Z>
I.i.d. noise, drop tokens	Thank you me to your party week .	for inviting last

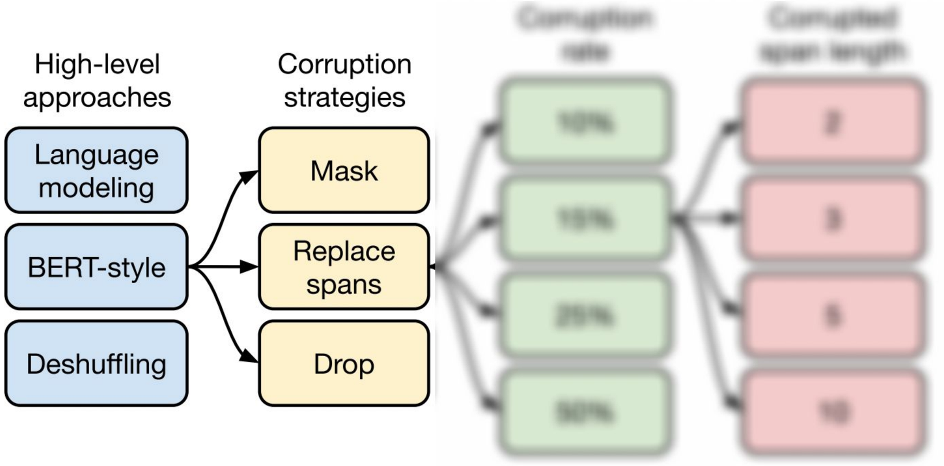
Different BERT-style Unsupervised Objectives



- Thank you <M> <M> me to your party <M> week . Thank you for inviting me to your party last week

MASS-style Song et al. (2019)	Thank you <M> <M> me to your party <M> week .	*(original text)*
I.i.d. noise, replace spans	Thank you <X> me to your party <Y> week .	<X> for inviting <Y> last <Z>
I.i.d. noise, drop tokens	Thank you me to your party week .	for inviting last

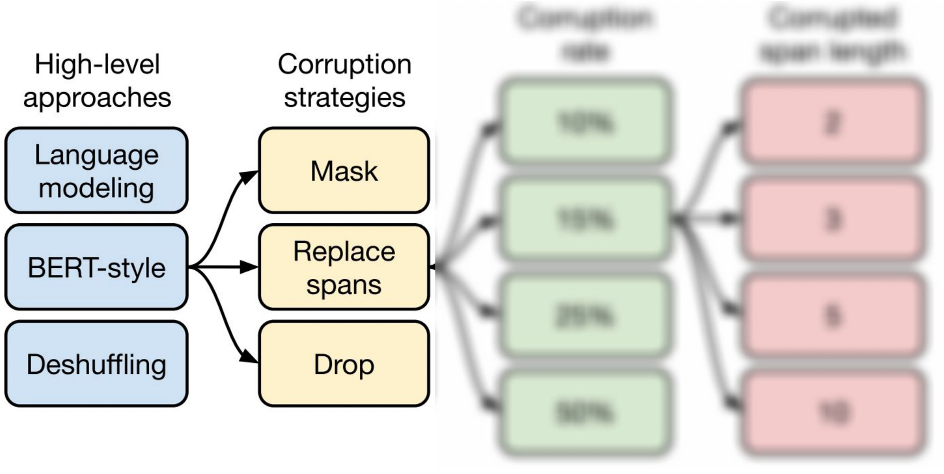
Different BERT-style Unsupervised Objectives



- Thank you <X> me to your party <Y> week . <X> for inviting <Y> last <Z>

MASS-style Song et al. (2019)	Thank you <M> <M> me to your party <M> week .	*(original text)*
I.i.d. noise, replace spans	Thank you <X> me to your party <Y> week .	<X> for inviting <Y> last <Z>
I.i.d. noise, drop tokens	Thank you me to your party week .	for inviting last

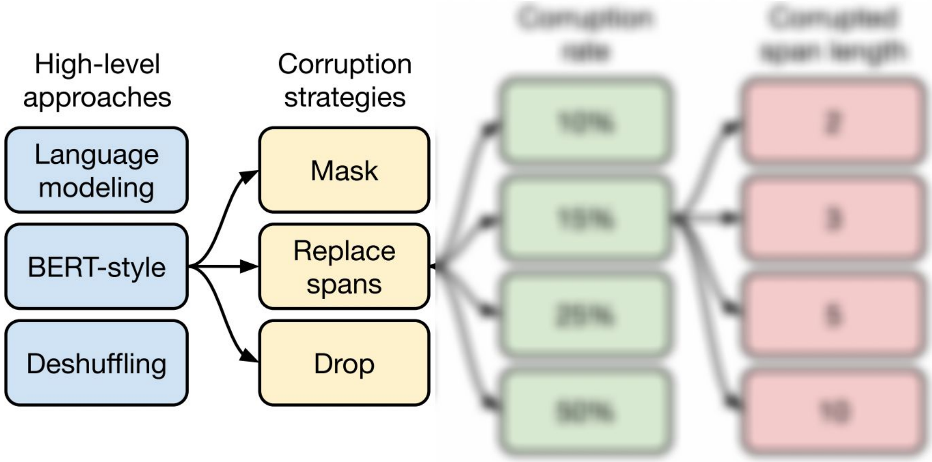
Different BERT-style Unsupervised Objectives



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

MASS-style Song et al. (2019)	Thank you <M> <M> me to your party <M> week .	*(original text)*
I.i.d. noise, replace spans	Thank you <X> me to your party <Y> week .	<X> for inviting <Y> last <Z>
I.i.d. noise, drop tokens	Thank you me to your party week .	for inviting last

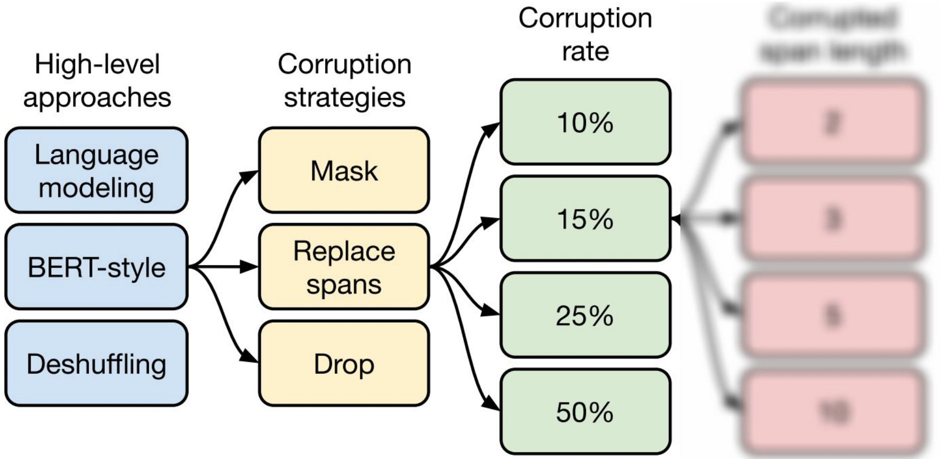
Comparison of variants of the BERT-style pre-training objective



1. All the variants perform similarly.
2. “Replace corrupted spans” and “Drop corrupted tokens” are more appealing because target sequences are shorter, speeding up training.

Objective	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
BERT-style (Devlin et al., 2018)	82.96	19.17	**80.65**	69.85	26.78	**40.03**	27.41
MASS-style (Song et al., 2019)	82.32	19.16	80.10	69.28	26.79	**39.89**	27.55
★ Replace corrupted spans	83.28	**19.24**	**80.88**	**71.36**	**26.98**	39.82	**27.65**
Drop corrupted tokens	**84.44**	**19.31**	**80.52**	68.67	**27.07**	39.76	**27.82**

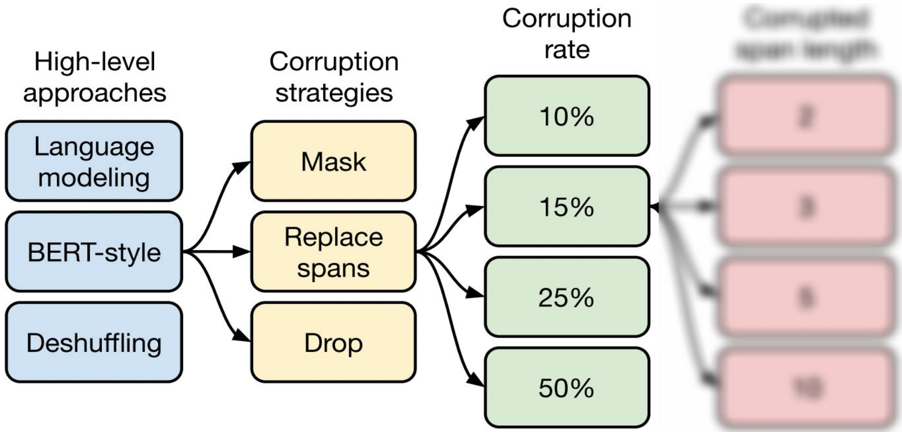
Different Corruption Rates



- Thank <X> for inviting me to <Y> party last week. <X> you <Y> your <Z>.
- Thank <X> for <Y> me to your party <Z>. <X> you <Y> inviting <Z> last week.

Objective	Inputs	Targets
I.i.d. noise, replace spans	Thank you <X> me to your party <Y> week .	<X> for inviting <Y> last <Z>

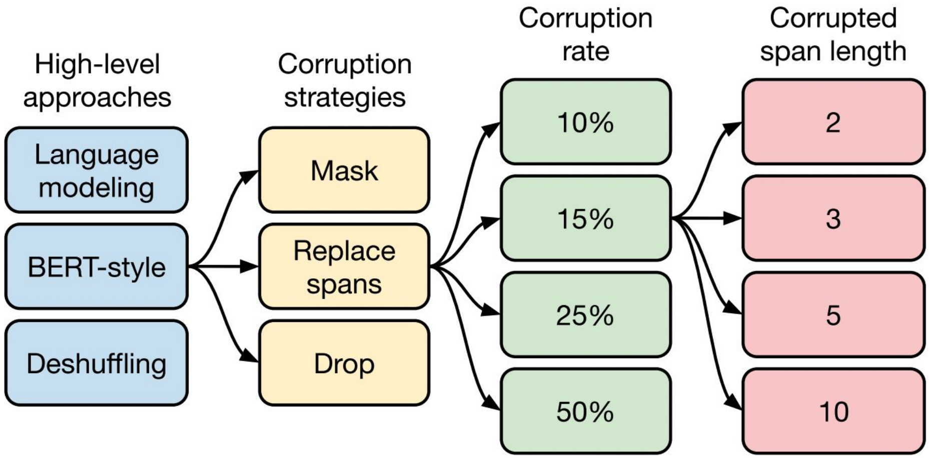
Performance of the i.i.d. corruption objective with different corruption rates



1. Larger corruption rate leads to downstream performance degradation.
2. Larger corruption rate also leads to longer targets, slowing down training.

Corruption rate	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
10%	**82.82**	19.00	**80.38**	69.55	**26.87**	39.28	**27.44**
★ 15%	**83.28**	19.24	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
25%	**83.00**	**19.54**	**80.96**	70.48	**27.04**	**39.83**	**27.47**
50%	81.27	19.32	79.80	70.33	**27.01**	**39.90**	**27.49**

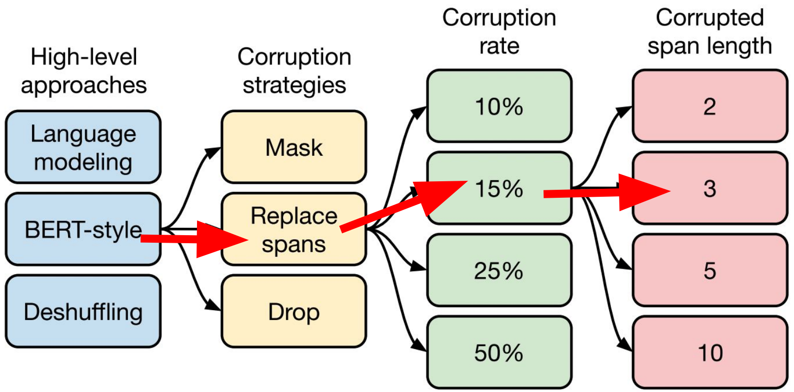
Different Corruption Rates



Objective	Inputs	Targets
Random spans	Thank you <X> to <Y> week .	<X> for inviting me <Y> your party last <Z>

- Thank <X> for inviting me to <Y> party last <Z>. <X> you <Y> your <Z> week.
    - Thank <X> me to your party week. <X> you for inviting <Z>.

Performance of the span-corruption objective for different average span lengths



1. Average span length of 3 works well on most non-translation tasks.
2. Span corruption produces shorter target sequences and leads to speedup in training.

Span length	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Baseline (i.i.d.)	**83.28**	19.24	80.88	71.36	**26.98**	**39.82**	**27.65**
2	**83.54**	19.39	**82.09**	**72.20**	**26.76**	**39.99**	**27.63**
3	**83.49**	**19.62**	**81.84**	**72.53**	**26.86**	39.65	**27.62**
5	**83.40**	19.24	**82.05**	**72.23**	**26.88**	39.40	**27.53**
10	82.85	19.33	**81.84**	70.44	**26.79**	39.49	**27.69**

Pre-training dataset

Performance from pre-training on different data sets.

Data set	Size	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ C4	745GB	83.28	**19.24**	80.88	71.36	**26.98**	**39.82**	**27.65**
C4, unfiltered	6.1TB	81.46	19.14	78.78	68.04	26.55	39.34	27.21
RealNews-like	35GB	**83.83**	**19.23**	80.39	2 72.38	**26.75**	**39.90**	**27.48**
WebText-like	17GB	**84.03**	**19.31**	**81.42**	71.40	**26.80**	**39.74**	**27.59**
Wikipedia	16GB	1 81.85	**19.31**	81.29	68.01	**26.94**	39.69	**27.67**
Wikipedia + TBC	20GB	83.65	**19.28**	**82.08**	3 **73.24**	**26.77**	39.63	**27.57**

Pre-training on in-domain data tends to help downstream task.

1 Much worse on COLA

2 Much better on ReCoRD

Question answering on News dataset

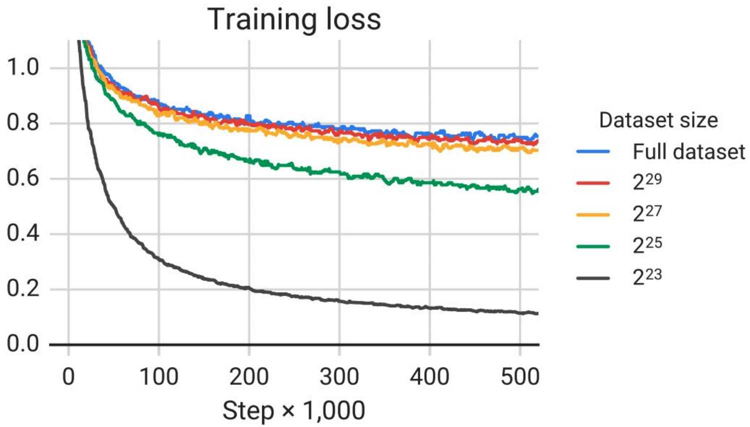
3 Much better on MultiRC

Question answering on Novel dataset

Check whether a sentence is linguistically correct?

Effect of repeating data during pre-training

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



1. Performance degrades as dataset size shrinks.
2. Model memorizes the pre-training data, with smaller dataset size.

Scaling

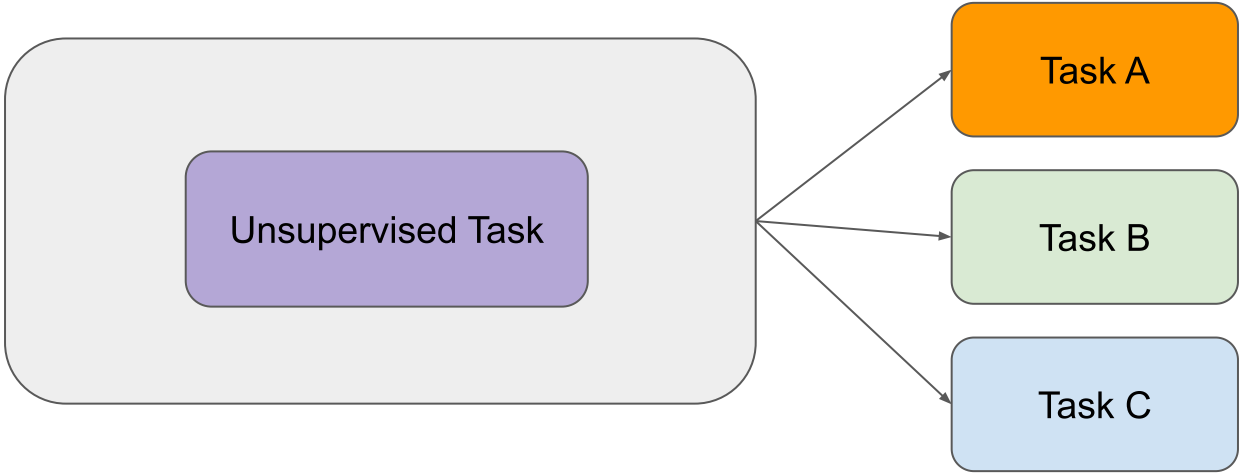
	Scaling strategy	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★	Baseline	83.28	19.24	80.88	71.36	26.98	39.82	27.65
	1× size, 4× training steps	85.33	19.33	82.45	74.72	27.08	40.66	27.93
	1× size, 4× batch size	84.60	19.42	82.52	74.64	27.07	40.60	27.84
	2× size, 2× training steps	**86.18**	19.66	**84.18**	77.18	27.52	**41.03**	28.19
	4× size, 1× training steps	**85.91**	19.73	**83.86**	**78.04**	27.47	40.71	28.10
	4× ensembled	84.77	**20.10**	83.09	71.74	**28.05**	40.53	**28.57**
	4× ensembled, fine-tune only	84.05	19.57	82.36	71.55	27.55	40.22	28.09

1. Advantage in increasing model size compared to simply increasing batch size or number of training steps.
2. Not much of a difference between increasing size + training and increasing size only
   a. Improving training time and model size are complementary means of improving performance.
3. Ensembling helps, except in SuperGLUE.

Multi-task training

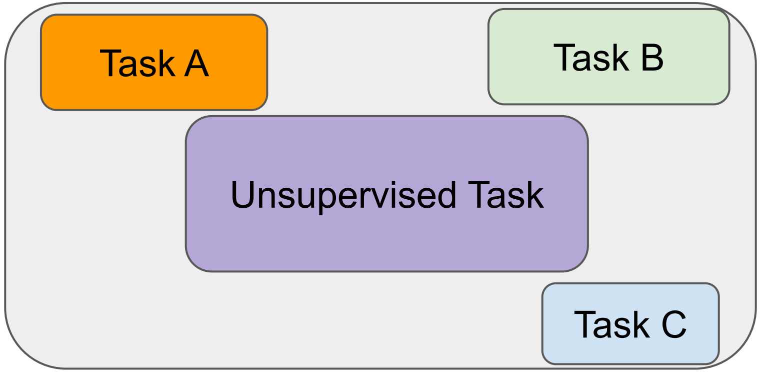
Multi-task

Training strategy	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Unsupervised pre-training + fine-tuning	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	39.82	27.65



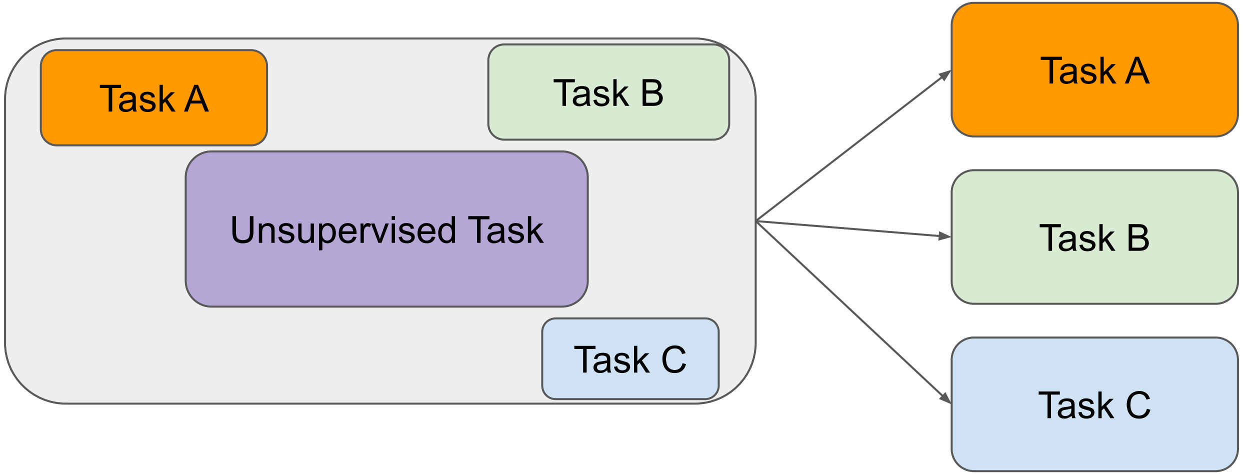
Multi-task

Training strategy	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Unsupervised pre-training + fine-tuning	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	39.82	27.65
Multi-task training	81.42	**19.24**	79.78	67.30	25.21	36.30	27.76



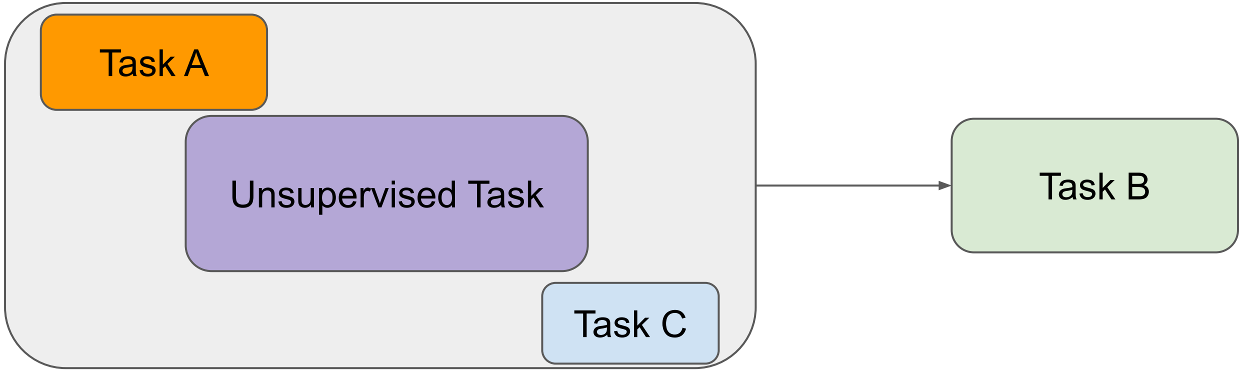
Multi-task

Training strategy	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Unsupervised pre-training + fine-tuning	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	39.82	27.65
Multi-task training	81.42	**19.24**	79.78	67.30	25.21	36.30	27.76
Multi-task pre-training + fine-tuning	**83.11**	**19.12**	**80.26**	**71.03**	**27.08**	39.80	**28.07**



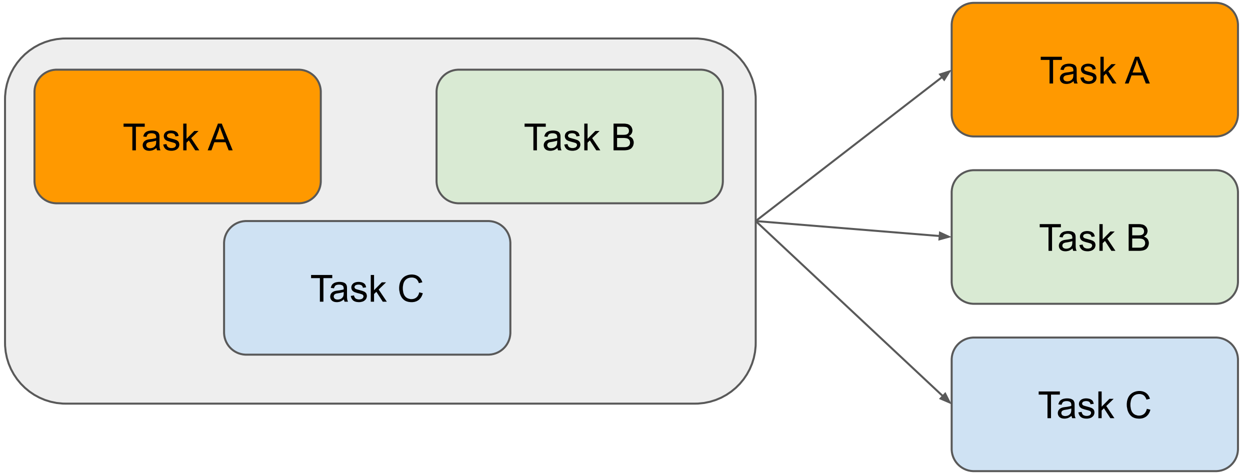
Multi-task

Training strategy	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Unsupervised pre-training + fine-tuning	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	39.82	27.65
Multi-task training	81.42	**19.24**	79.78	67.30	25.21	36.30	27.76
Multi-task pre-training + fine-tuning	**83.11**	**19.12**	**80.26**	**71.03**	**27.08**	39.80	**28.07**
Leave-one-out multi-task training	81.98	19.05	79.97	**71.68**	**26.93**	39.79	**27.87**



Multi-task

Training strategy	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Unsupervised pre-training + fine-tuning	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	39.82	27.65
Multi-task training	81.42	**19.24**	79.78	67.30	25.21	36.30	27.76
Multi-task pre-training + fine-tuning	**83.11**	**19.12**	**80.26**	**71.03**	**27.08**	39.80	**28.07**
Leave-one-out multi-task training	81.98	19.05	79.97	**71.68**	**26.93**	39.79	**27.87**
Supervised multi-task pre-training	79.93	18.96	77.38	65.36	26.81	**40.13**	**28.04**



Multi-task

Training strategy	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Unsupervised pre-training + fine-tuning	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	39.82	27.65
Multi-task training	81.42	**19.24**	79.78	67.30	25.21	36.30	27.76
Multi-task pre-training + fine-tuning	**83.11**	**19.12**	**80.26**	**71.03**	**27.08**	39.80	**28.07**
Leave-one-out multi-task training	81.98	19.05	79.97	**71.68**	**26.93**	39.79	**27.87**
Supervised multi-task pre-training	79.93	18.96	77.38	65.36	26.81	**40.13**	**28.04**

1. Multi-task pre-training + fine-tuning works as well as unsupervised pre-training + fine-tuning.
2. Practical benefit of Multi-task pre-training + fine-tuning is to monitor downstream performance during pre-training.

Putting it all together

Encoder-decoder architecture

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

Span prediction objective

Span length	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Baseline (i.i.d.)	**83.28**	19.24	80.88	71.36	**26.98**	**39.82**	**27.65**
2	**83.54**	19.39	**82.09**	**72.20**	**26.76**	**39.99**	**27.63**
3	**83.49**	**19.62**	**81.84**	**72.53**	**26.86**	39.65	**27.62**
5	**83.40**	19.24	**82.05**	**72.23**	**26.88**	39.40	**27.53**
10	82.85	19.33	**81.84**	70.44	**26.79**	39.49	**27.69**

C4 dataset

Data set	Size	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ C4	745GB	83.28	**19.24**	80.88	71.36	**26.98**	**39.82**	**27.65**
C4, unfiltered	6.1TB	81.46	19.14	78.78	68.04	26.55	39.34	27.21
RealNews-like	35GB	**83.83**	**19.23**	80.39	72.38	**26.75**	**39.90**	**27.48**
WebText-like	17GB	**84.03**	**19.31**	**81.42**	71.40	**26.80**	**39.74**	**27.59**
Wikipedia	16GB	81.85	**19.31**	81.29	68.01	**26.94**	39.69	**27.67**
Wikipedia + TBC	20GB	83.65	**19.28**	**82.08**	**73.24**	**26.77**	39.63	**27.57**

Multi-task pre-training

Training strategy	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Unsupervised pre-training + fine-tuning	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	39.82	27.65
Multi-task training	81.42	**19.24**	79.78	67.30	25.21	36.30	27.76
Multi-task pre-training + fine-tuning	**83.11**	**19.12**	**80.26**	**71.03**	**27.08**	39.80	**28.07**
Leave-one-out multi-task training	81.98	19.05	79.97	**71.68**	**26.93**	39.79	**27.87**
Supervised multi-task pre-training	79.93	18.96	77.38	65.36	26.81	**40.13**	**28.04**

Bigger model trained longer

Scaling strategy	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Baseline	83.28	19.24	80.88	71.36	26.98	39.82	27.65
1× size, 4× training steps	85.33	19.33	82.45	74.72	27.08	40.66	27.93
1× size, 4× batch size	84.60	19.42	82.52	74.64	27.07	40.60	27.84
2× size, 2× training steps	**86.18**	19.66	**84.18**	77.18	27.52	**41.03**	28.19
4× size, 1× training steps	**85.91**	19.73	**83.86**	**78.04**	27.47	40.71	28.10
4× ensembled	84.77	**20.10**	83.09	71.74	**28.05**	40.53	**28.57**
4× ensembled, fine-tune only	84.05	19.57	82.36	71.55	27.55	40.22	28.09

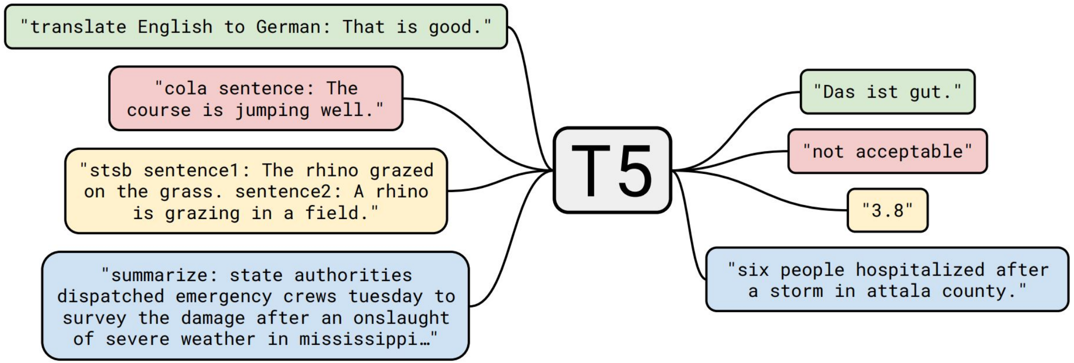
Model Variants

Model	Parameters	No. of layers	d_{model}	d_{ff}	d_{kv}	No. of heads
Small	60M	6	512	2048	64	8
Base	220M	12	768	3072	64	12
Large	770M	24	1024	4096	64	16
3B	3B	24	1024	16384	128	32
11B	11B	24	1024	65536	128	128

Model	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
Previous best	89.4	20.30	95.5	84.6	**33.8**	**43.8**	**38.5**
T5-Small	77.4	19.56	87.24	63.3	26.7	36.0	26.8
T5-Base	82.7	20.34	92.08	76.2	30.9	41.2	28.0
T5-Large	86.4	20.68	93.79	82.3	32.0	41.5	28.1
T5-3B	88.5	21.02	94.95	86.4	31.8	42.6	28.2
T5-11B	**89.7**	**21.55**	**95.64**	**88.9**	32.1	43.4	28.1

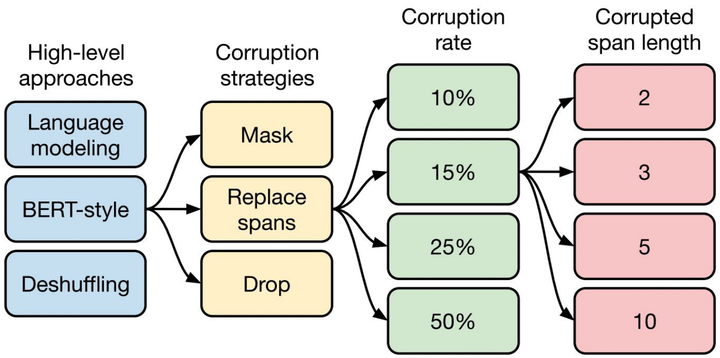
Let's review!

- **Unified text-to-text framework**
- Supports both discriminative and generative tasks
  - Classification, summarization, translation, etc.
  - Better on GLUE/SuperGLUE, SQuAD, and summarization; less on translation



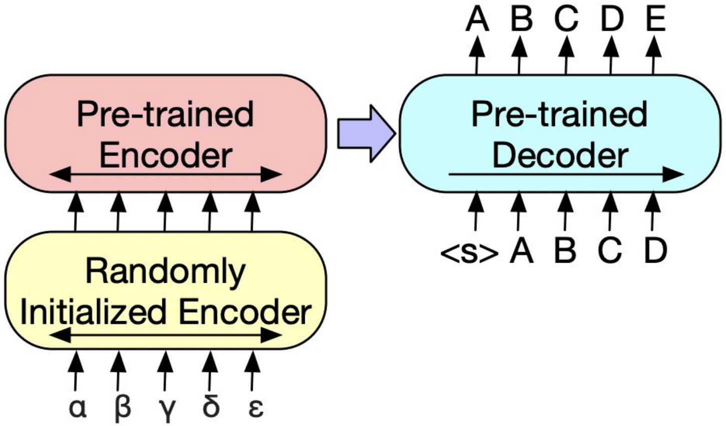
Let’s review!

- “Empirical comparison of existing techniques”
  - Evidence for encoder-decoder models, span masking, multi-task pre-training
- Still no limit on large model improvements?
- C4 as a *large*, clean corpus



Other Variants

- BART (Lewis et al. 2020)
  - Similar Architecture as T5.
  - Performs competitive to RoBERTa and XLNet on discriminative tasks.
  - Outperformed existing methods on question answering, and summarization tasks.
  - Improved results on machine translation with fine-tuning on target language.
- mT5 (Xue et al. 2021)
  - Discussed earlier!



Other Variants

- AlexaTM 20B (Soltan et al. 2022)
  - Larger architecture on multilingual C4 dataset.
  - Can outperform much larger autoregressive models (GPT-3 175B) in zero shot tasks.

Model	BoolQ (acc)	CB (acc)	RTE (acc)	ReCoRD (acc)	WSC (acc)	WiC (acc)	CoPA (acc)	MultiRC (f1a)	Avg
PaLM 540B	**88.0**	51.8	**72.9**	**92.9**	**89.1**	**59.1**	**93.0**	**83.5**	**78.8**
GPT3 175B	60.5	46.4	63.5	90.2	65.4	0.0	91.0	72.9	61.2
BLOOM 175B	63.5	33.9	52.0	NA	51.9	50.6	56.0	57.1	NA
GPT3 13B	66.2	19.6	62.8	89.0	64.4	0.0	84.0	71.4	57.2
UL 20B	63.1	41.1	60.7	88.1	79.9	49.8	85.0	36.2	63.0
AlexaTM 20B	69.44	**67.9**	68.59	88.4	68.27	53.29	78.0	59.57	69.16

Q1. Describe how T5 is adapted to sentence classification tasks

[Task-specific prefix]: [Input text]

- CoLA (GLUE; Classification):
  "cola sentence: The course is jumping well." -> "not acceptable"
- STS-B (GLUE; Regression):
  "stsb sentence1: The rhino grazed. sentence2: A rhino is grazing." -> "3.8"

"cola sentence: The course is jumping well." -> "hamburger"

"Hamburger" is not a valid CoLA output, so this is a fail!

Q2. Can you think of a reason why generating the entire output performs worse than only generating the masked spans?

Objective	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
BERT-style (Devlin et al., 2018)	82.96	19.17	**80.65**	69.85	26.78	**40.03**	27.41
MASS-style (Song et al., 2019)	82.32	19.16	80.10	69.28	26.79	**39.89**	27.55
★ Replace corrupted spans	83.28	**19.24**	**80.88**	**71.36**	**26.98**	39.82	**27.65**
Drop corrupted tokens	**84.44**	**19.31**	**80.52**	68.67	**27.07**	39.76	**27.82**

Q3. Would you expect a <u>BERT encoder or a T5 encoder</u> to learn <u>richer linguistic features</u>, assuming both were the same size and trained for the same number of steps? How would it change if the <u>average masked span length was increased</u>?