T5 (and encoder-decoder models)

THE UNIVERSITY OF UTAH

Slides credit: Daniel Kashabi, Collin Raffel, Abhishek Panigrahi, Victoria Graf and others

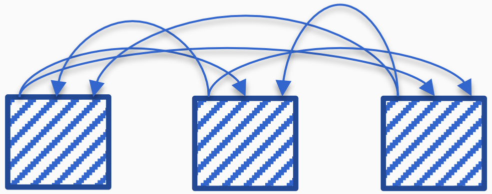
Outline

- Encoders, decoders and encoder-decoders
- What is T5?
- Design choices

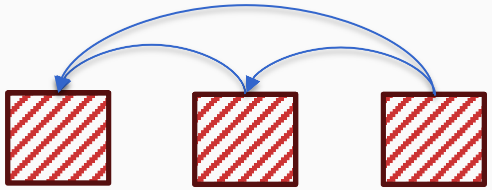
Outline

- Encoders, decoders and encoder-decoders
- What is T5?
- Design choices

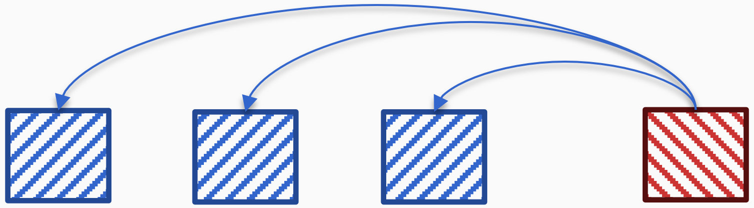
Three different kinds of attention



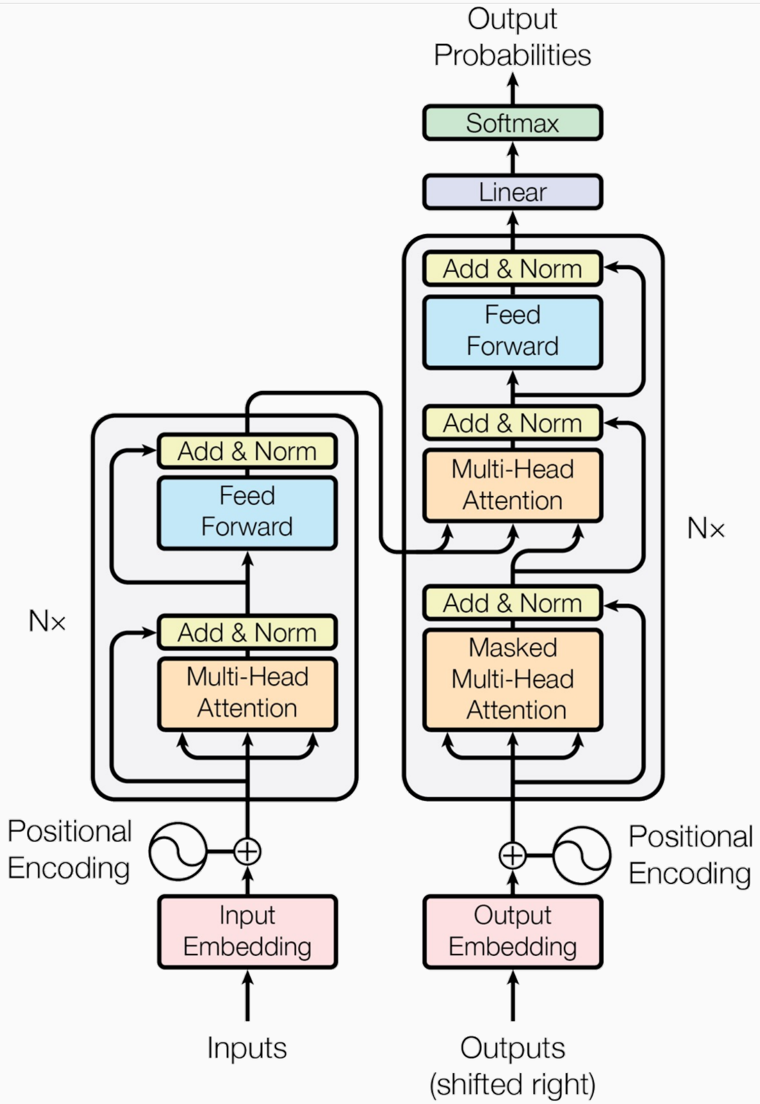
Encoder self-attention



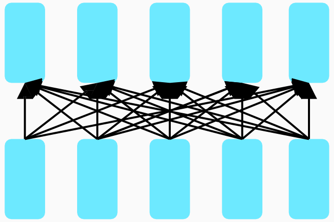
Masked decoder self-attention



Encoder-decoder self-attention



Transformers are the default building blocks for NLP today

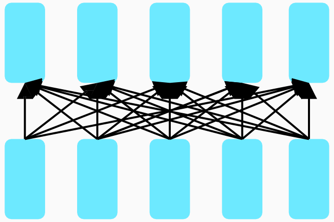


Encoders

Examples: BERT, RoBERTa, SciBERT.

Captures bidirectional context

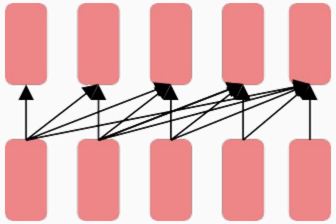
Transformers are the default building blocks for NLP today



Encoders

Examples: BERT, RoBERTa, SciBERT.

Captures bidirectional context



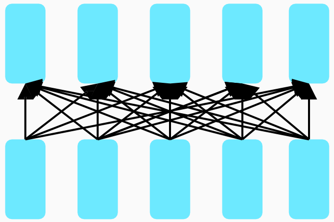
Decoders

Examples: GPT-2, GPT-3, LaMDA

Also known as: causal or auto-regressive language model

Natural if the goal is generation, but can not condition on future words

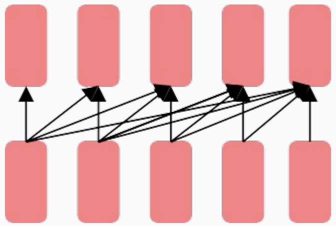
Transformers are the default building blocks for NLP today



Encoders

Examples: BERT, RoBERTa, SciBERT.

Captures bidirectional context

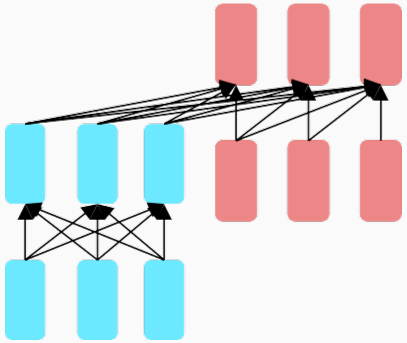


Decoders

Examples: GPT-2, GPT-3, LaMDA

Also known as: causal or auto-regressive language model

Natural if the goal is generation, but can not condition on future words

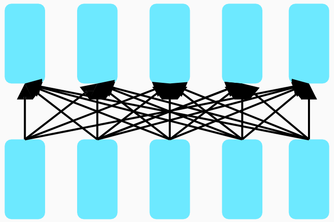


Encoder-Decoders

Examples: BART, T5, Meena

Conditional generation based on an encoded input

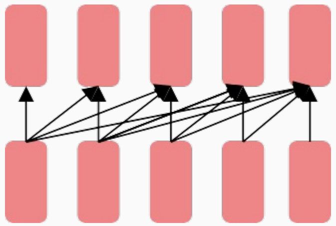
Transformers are the default building blocks for NLP today



Encoders

Examples: BERT, RoBERTa, SciBERT.

Captures bidirectional context

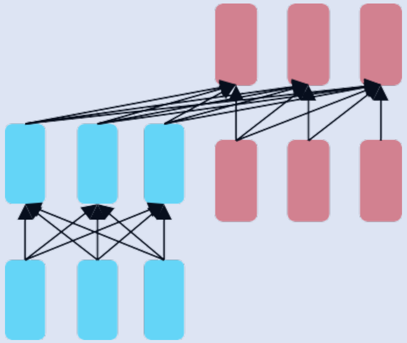


Decoders

Examples: GPT-2, GPT-3, LaMDA

Also known as: causal or auto-regressive language model

Natural if the goal is generation, but can not condition on future words



Encoder-Decoders

Examples: BART, T5, Meena

Conditional generation based on an encoded input

This lecture

Outline

- Encoders, decoders and encoder-decoders
- What is T5?
- Design choices

T5: Text-To-Text Transfer Transformer

[Raffel et al 2019]

This paper:

- Represent a collection of NLP tasks in a common format that takes in text and produces text

- An encoder decoder architecture

- A thorough exploration of model design choices

Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer

Colin Raffel* CRAFFEL@GMAIL.COM
Noam Shazeer* NOAM@GOOGLE.COM
Adam Roberts* ADAROB@GOOGLE.COM
Katherine Lee* KATHERINELEE@GOOGLE.COM
Sharan Narang SHARANNARANG@GOOGLE.COM
Michael Matena MMATENA@GOOGLE.COM
Yanqi Zhou YANQIZ@GOOGLE.COM
Wei Li MWEILI@GOOGLE.COM
Peter J. Liu PETERJLIU@GOOGLE.COM
Google, Mountain View, CA 94043, USA

Editor: Ivan Titov

Abstract

Transfer learning, where a model is first pre-trained on a data-rich task before being fine-tuned on a downstream task, has emerged as a powerful technique in natural language processing (NLP). The effectiveness of transfer learning has given rise to a diversity of approaches, methodology, and practice. In this paper, we explore the landscape of transfer learning techniques for NLP by introducing a unified framework that converts all text-based language problems into a text-to-text format. Our systematic study compares pre-training objectives, architectures, unlabeled data sets, transfer approaches, and other factors on dozens of language understanding tasks. By combining the insights from our exploration with scale and our new "Colossal Clean Crawled Corpus", we achieve state-of-the-art results on many benchmarks covering summarization, question answering, text classification, and more. To facilitate future work on transfer learning for NLP, we release our data set, pre-trained models, and code.[1]

Keywords: transfer learning, natural language processing, multi-task learning, attention-based models, deep learning

T5: Text-To-Text Transfer Transformer

[Raffel et al 2019]

This paper:

Represent a collection of NLP tasks in a common format that takes in text and produces text

An encoder decoder architecture

A thorough exploration of model design choices

Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer

Colin Raffel* CRAFFEL@GMAIL.COM
Noam Shazeer* NOAM@GOOGLE.COM
Adam Roberts* ADAROB@GOOGLE.COM
Katherine Lee* KATHERINELEE@GOOGLE.COM
Sharan Narang SHARANNARANG@GOOGLE.COM
Michael Matena MMATENA@GOOGLE.COM
Yanqi Zhou YANQIZ@GOOGLE.COM
Wei Li MWEILI@GOOGLE.COM
Peter J. Liu PETERJLIU@GOOGLE.COM

Google, Mountain View, CA 94043, USA

Editor: Ivan Titov

Abstract

Transfer learning, where a model is first pre-trained on a data-rich task before being fine-tuned on a downstream task, has emerged as a powerful technique in natural language processing (NLP). The effectiveness of transfer learning has given rise to a diversity of approaches, methodology, and practice. In this paper, we explore the landscape of transfer learning techniques for NLP by introducing a unified framework that converts all text-based language problems into a text-to-text format. Our systematic study compares pre-training objectives, architectures, unlabeled data sets, transfer approaches, and other factors on dozens of language understanding tasks. By combining the insights from our exploration with scale and our new "Colossal Clean Crawled Corpus", we achieve state-of-the-art results on many benchmarks covering summarization, question answering, text classification, and more. To facilitate future work on transfer learning for NLP, we release our data set, pre-trained models, and code.[1]

Keywords: transfer learning, natural language processing, multi-task learning, attention-based models, deep learning

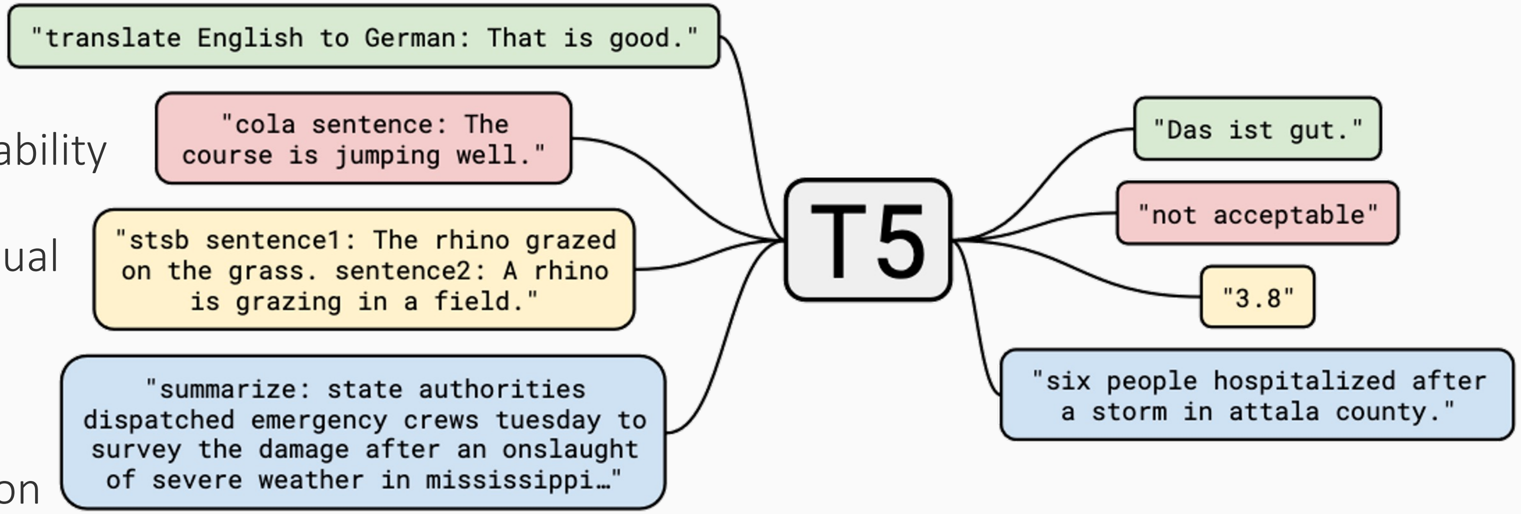
The claim: All text processing tasks → text-to-text format

Translation

Linguistic acceptability

Semantic textual similarity

Summarization



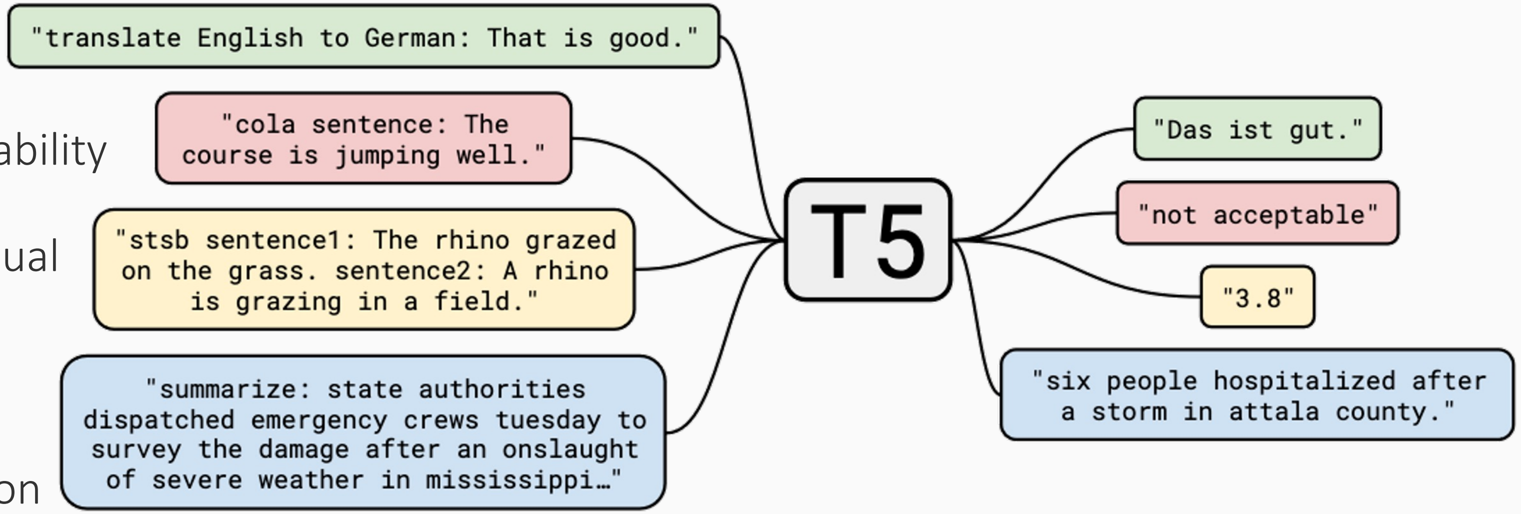
The claim: All text processing tasks → text-to-text format

Translation

Linguistic acceptability

Semantic textual similarity

Summarization



Textual entailment
Paraphrase recognition
Reading comprehension
…

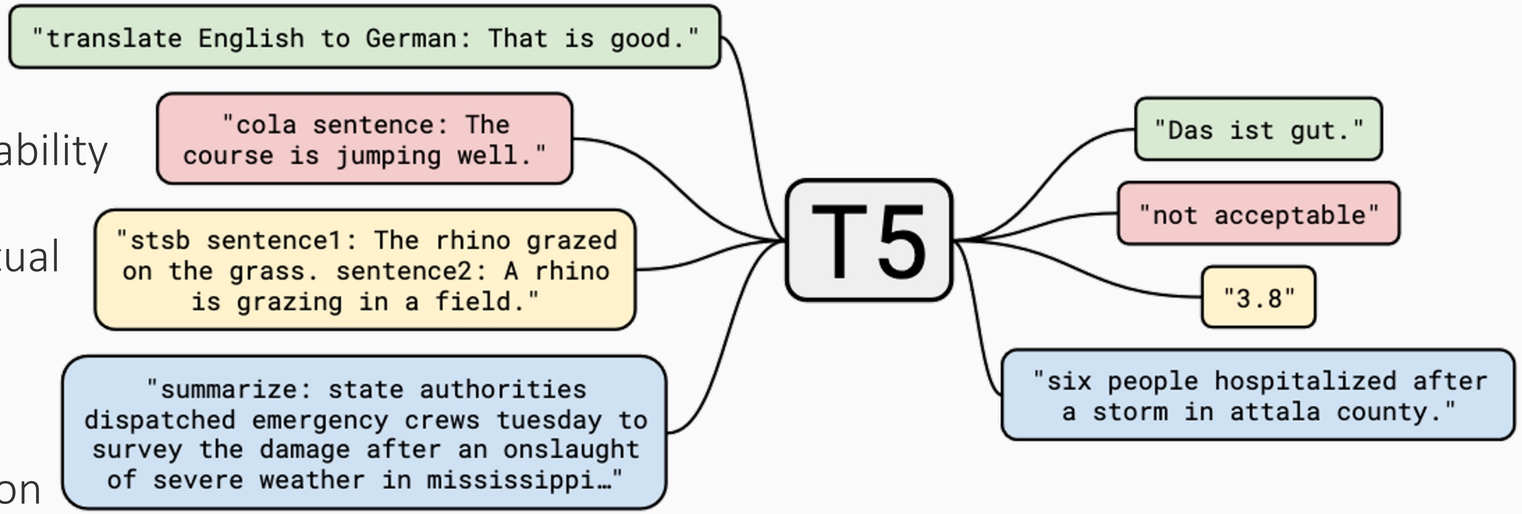
The claim: All text processing tasks → text-to-text format

Translation

Linguistic acceptability

Semantic textual similarity

Summarization



Textual entailment
Paraphrase recognition
Reading comprehension
…

For each task, design a template so that the input and outputs are text

(Some previous papers had also explored this idea)

Outline

- Encoders, decoders and encoder-decoders
- What is T5?
- Design choices

T5: Text-To-Text Transfer Transformer

[Raffel et al 2019]

This paper:

- Represent a collection of NLP tasks in a common format that takes in text and produces text
- An encoder decoder architecture
- A thorough exploration of model design choices

Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer

Colin Raffel* CRAFFEL@GMAIL.COM
Noam Shazeer* NOAM@GOOGLE.COM
Adam Roberts* ADAROB@GOOGLE.COM
Katherine Lee* KATHERINELEE@GOOGLE.COM
Sharan Narang SHARANNARANG@GOOGLE.COM
Michael Matena MMATENA@GOOGLE.COM
Yanqi Zhou YANQIZ@GOOGLE.COM
Wei Li MWEILI@GOOGLE.COM
Peter J. Liu PETERJLIU@GOOGLE.COM
Google, Mountain View, CA 94043, USA

Editor: Ivan Titov

Abstract

Transfer learning, where a model is first pre-trained on a data-rich task before being fine-tuned on a downstream task, has emerged as a powerful technique in natural language processing (NLP). The effectiveness of transfer learning has given rise to a diversity of approaches, methodology, and practice. In this paper, we explore the landscape of transfer learning techniques for NLP by introducing a unified framework that converts all text-based language problems into a text-to-text format. Our systematic study compares pre-training objectives, architectures, unlabeled data sets, transfer approaches, and other factors on dozens of language understanding tasks. By combining the insights from our exploration with scale and our new "Colossal Clean Crawled Corpus", we achieve state-of-the-art results on many benchmarks covering summarization, question answering, text classification, and more. To facilitate future work on transfer learning for NLP, we release our data set, pre-trained models, and code.[1]

Keywords: transfer learning, natural language processing, multi-task learning, attention-based models, deep learning

Numerous model design choices affect performance

- What is the model architecture?
- What is the right pre-training objective
- Which data should we use for pre-training?
- How much pre-training?
- Fine tune on one task? Fine tune on multiple tasks? Some combination?
- How big should the model be?

Numerous model design choices affect performance

- What is the model architecture?
- What is the right pre-training objective
- Which data should we use for pre-training?
- How much pre-training?
- Fine tune on one task? Fine tune on multiple tasks? Some combination?
- How big should the model be?

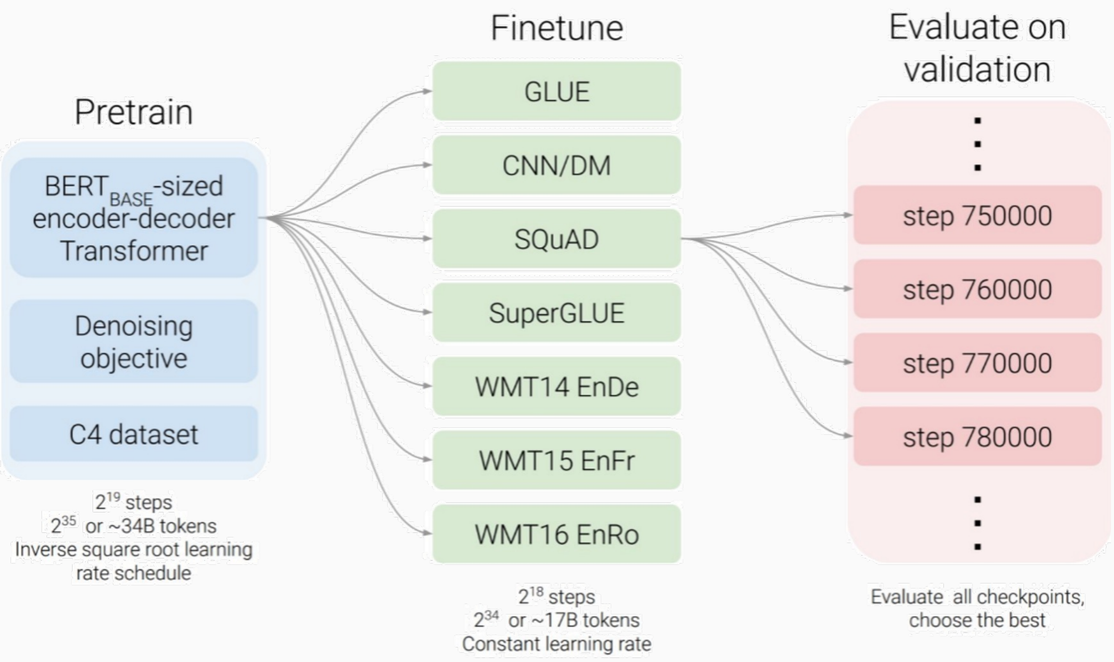
Can we understand the impact of each choice
by altering it while keeping other choices fixed?

Experimental Setup

Decide a default model

- Encoder-decoder architecture
- Pretraining objective
- ….

Evaluate a design axis, fixing the rest of the parameters



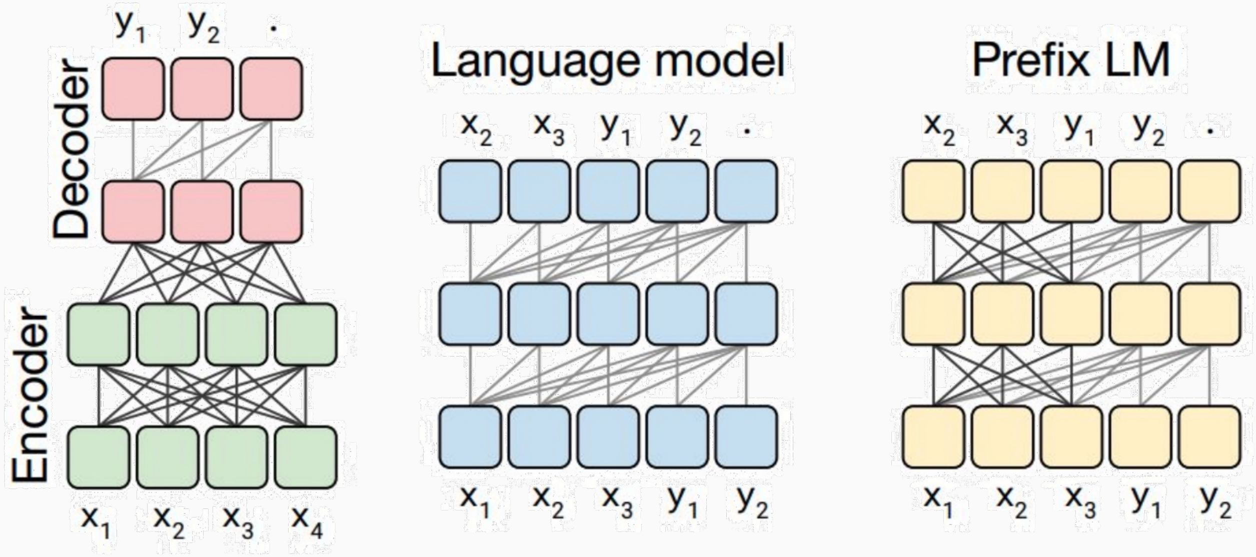
Key findings

Model Architectures	Encoder-decoder models outperform "decoder-only" language models
Pre-training Objectives	Fill-in-the-blank-style denoising objectives are most effective. Computational cost is a crucial factor
Unlabeled Datasets	Training on in-domain data is beneficial, but pre-training on smaller datasets can lead to overfitting
Training Strategies	Multitask learning is competitive with pre-train-then-fine-tune, but task frequency needs careful consideration
Scale	Comparison of scaling up model size, training time, and ensembled models for optimal use of fixed compute power

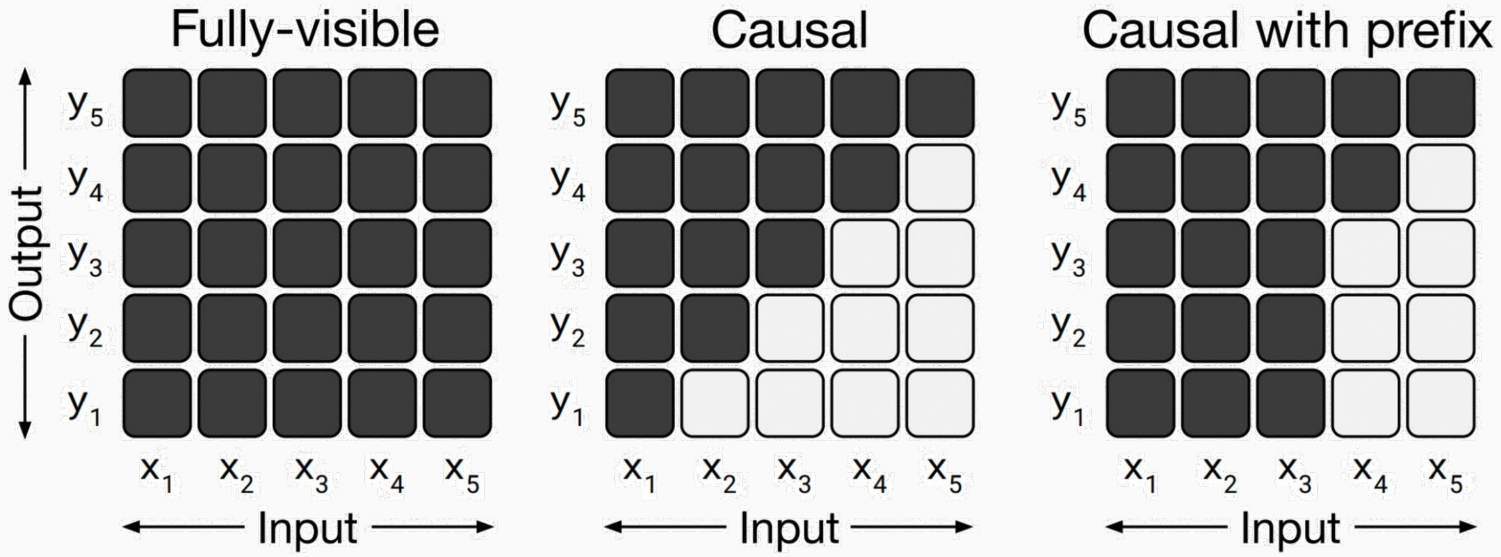
Key findings

Model Architectures	Encoder-decoder models outperform "decoder-only" language models
Pre-training Objectives	Fill-in-the-blank-style denoising objectives are most effective. Computational cost is a crucial factor
Unlabeled Datasets	Training on in-domain data is beneficial, but pre-training on smaller datasets can lead to overfitting
Training Strategies	Multitask learning is competitive with pre-train-then-fine-tune, but task frequency needs careful consideration
Scale	Comparison of scaling up model size, training time, and ensembled models for optimal use of fixed compute power

Architectures: Different Choices



Architectures: Different Attention Masks



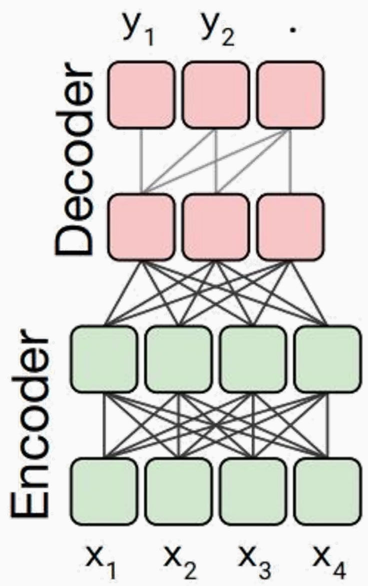
Allows the self attention mechanism to attend to the full input.

Doesn't allow output elements to look into the future

Allows to fully-visible masking on a portion of input

Architectural Variants: Experiments

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**

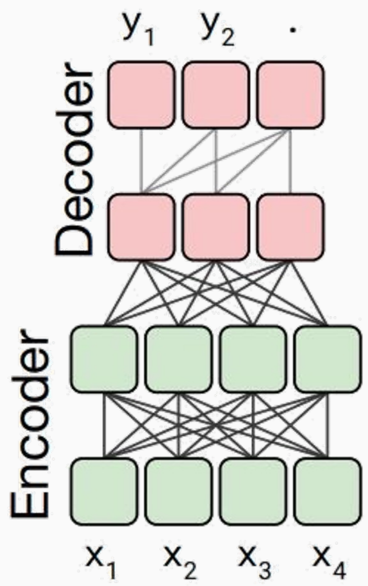


Slide credit: Abhishek Panigrahi, Victoria Graf

Architectural Variants: Experiments

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**

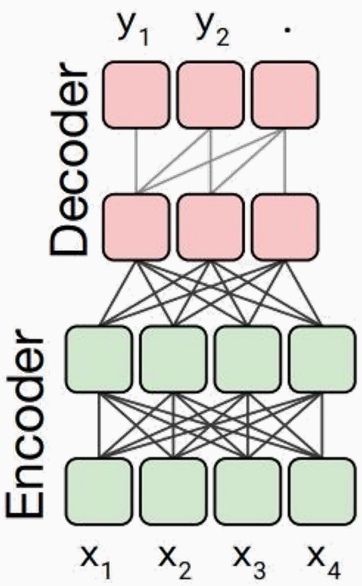
Input: Thank you for <X> me to your party <Y>.
Target: <X> inviting <Y> last week.



Architectural Variants: Experiments

	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

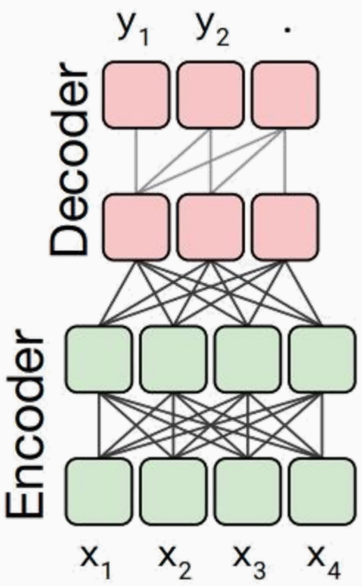


Slide credit: Abhishek Panigrahi, Victoria Graf

Architectural Variants: Experiments

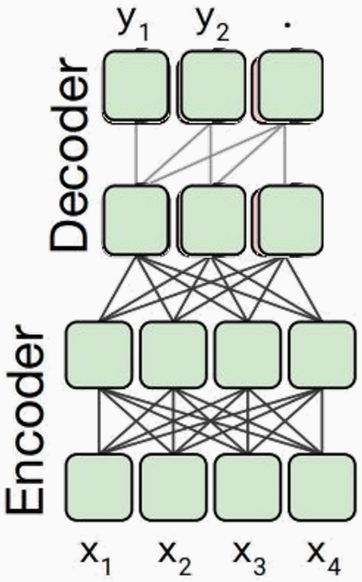
Architecture	Objective	Params	Cost	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Encoder-decoder	Denoising	$2P$	M	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**

Number of flops



Architectural Variants: Experiments

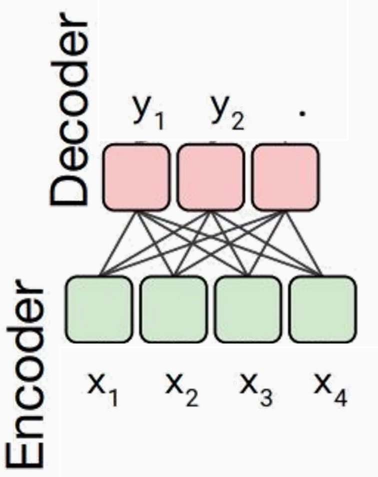
	Architecture	Objective	Params	Cost	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★	Encoder-decoder	Denoising	$2P$	M	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
	Enc-dec, shared	Denoising	P	M	82.81	18.78	**80.63**	**70.73**	26.72	39.03	**27.46**



Slide credit: Abhishek Panigrahi, Victoria Graf

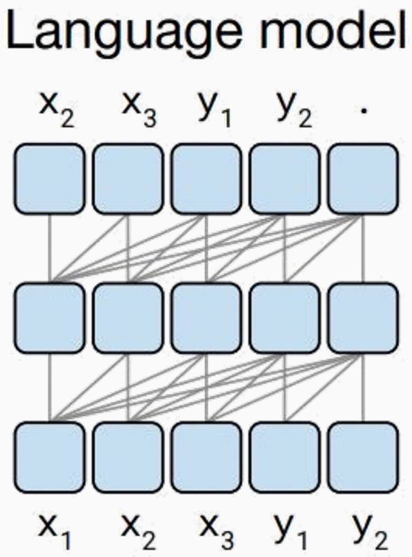
Architectural Variants: Experiments

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



Architectural Variants: Experiments

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

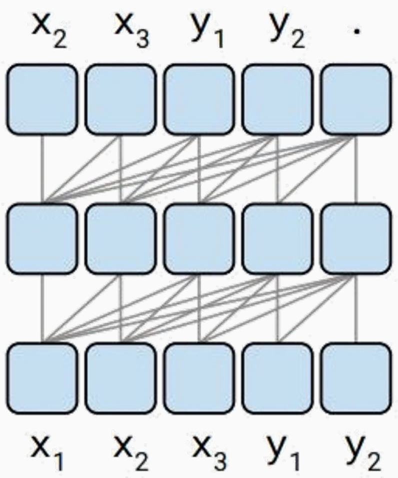


Architectural Variants: Experiments

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

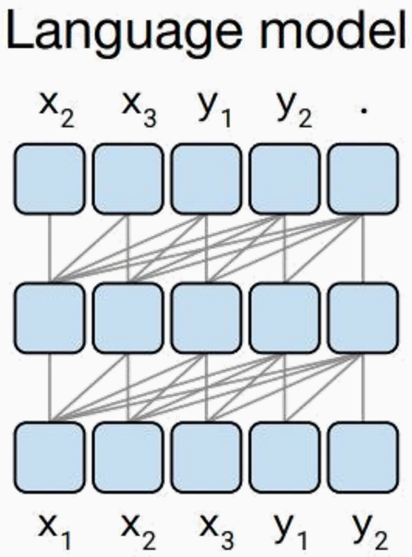
Language model is decoder-only

Language model



Architectural Variants: Experiments

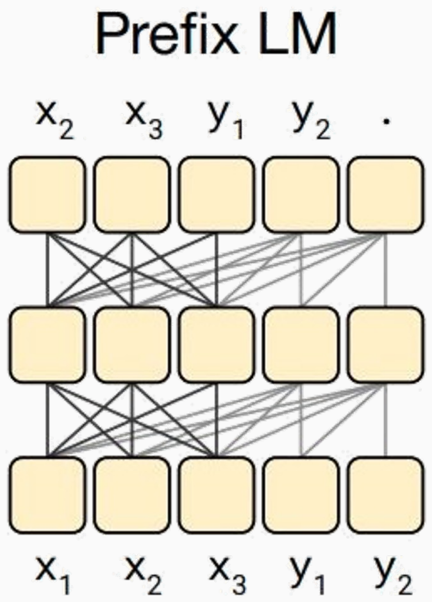
Architecture	Objective	Params	Cost	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Encoder-decoder	Denoising	$2P$	M	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
Enc-dec, shared	Denoising	P	M	82.81	18.78	**80.63**	**70.73**	26.72	39.03	**27.46**
Enc-dec, 6 layers	Denoising	P	$M/2$	80.88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

Architectural Variants: Experiments

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



Slide credit: Abhishek Panigrahi, Victoria Graf

Architectural Variants: Experiments

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

– Halving the number of layers in encoder and decoder hurts the performance.

– Performance of Encoder and Decoder with shared parameters is better than decoder only LM and prefix LM.

Slide credit: Abhishek Panigrahi, Victoria Graf

Key findings

Model Architectures	Encoder-decoder models outperform "decoder-only" language models
Pre-training Objectives	Fill-in-the-blank-style denoising objectives are most effective. Computational cost is a crucial factor
Unlabeled Datasets	Training on in-domain data is beneficial, but pre-training on smaller datasets can lead to overfitting
Training Strategies	Multitask learning is competitive with pre-train-then-fine-tune, but task frequency needs careful consideration
Scale	Comparison of scaling up model size, training time, and ensembled models for optimal use of fixed compute power

Pretraining objectives

The paper considered multiple different kinds of pre-training objectives

The research question: What training objective is best for self-supervised pre-training?

Pretraining objectives

The paper considered multiple different kinds of pre-training objectives

Objective	Example input	Example output
Prefix language modeling	Thank you for inviting	me to your party last week

Pretraining objectives

The paper considered multiple different kinds of pre-training objectives

Objective	Example input	Example output
Prefix language modeling	Thank you for inviting	me to your party last week
BERT-style denoising	Thank you <M> <M> me to your party apple week .	Thank you for inviting me to your party last week

Pretraining objectives

The paper considered multiple different kinds of pre-training objectives

Objective	Example input	Example output
Prefix language modeling	Thank you for inviting	me to your party last week
BERT-style denoising	Thank you <M> <M> me to your party apple week .	Thank you for inviting me to your party last week
Deshuffling	party me for your to . last fun you inviting week Thank	Thank you for inviting me to your party last week

Pretraining objectives

The paper considered multiple different kinds of pre-training objectives

Objective	Example input	Example output
Prefix language modeling	Thank you for inviting	me to your party last week
BERT-style denoising	Thank you <M> <M> me to your party apple week .	Thank you for inviting me to your party last week
Deshuffling	party me for your to . last fun you inviting week Thank	Thank you for inviting me to your party last week
I.i.d. noise, replace spans	Thank you <X> me to your party <Y> week .	<X> for inviting <Y> last <Z>

Pretraining objectives

The paper considered multiple different kinds of pre-training objectives

Objective	Example input	Example output
Prefix language modeling	Thank you for inviting	me to your party last week
BERT-style denoising	Thank you <M> <M> me to your party apple week .	Thank you for inviting me to your party last week
Deshuffling	party me for your to . last fun you inviting week Thank	Thank you for inviting me to your party last week
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

Comparing pre-training objectives

All the variants perform similarly

"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
Prefix language modeling	80.69	18.94	77.99	65.27	**26.86**	39.73	**27.49**
Deshuffling	73.17	18.59	67.61	58.47	26.11	39.30	25.62
BERT-style (Devlin et al., 2018)	82.96	19.17	**80.65**	69.85	26.78	**40.03**	27.41
★ 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**

How much data corruption is good enough?

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

- Little corruption rate may prevent effective learning.
- Larger corruption rate leads to downstream performance degradation.
- 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**

Key findings

Model Architectures	Encoder-decoder models outperform "decoder-only" language models
Pre-training Objectives	Fill-in-the-blank-style denoising objectives are most effective. Computational cost is a crucial factor
Unlabeled Datasets	Training on in-domain data is beneficial, but pre-training on smaller datasets can lead to overfitting
Training Strategies	Multitask learning is competitive with pre-train-then-fine-tune, but task frequency needs careful consideration
Scale	Comparison of scaling up model size, training time, and ensembled models for optimal use of fixed compute power

C4: Colossal Clean Crawled Corpus

Web-extracted text from April 2019

- English language only (`langdetect`)
- 750GB

Retains

- Sentences with terminal punctuation marks
- Only one copy of three sentence spans that occur more than once

Removes

- Pages with fewer than 5 sentences
- Sentences with fewer than 3 words
- References to Javascript
- Placeholder “Lorem ipsum” text
- Obsceneties

Play with the data: `https://c4-search.apps.allenai.org/`

C4: Colossal Clean Crawled Corpus

Web-extracted text from April 2019
- English language only (`langdetect`)
- 750GB

Retains
- Sentences with terminal punctuation marks
- Only one copy of three sentence spans that occur more than once

Removes
- Pages with fewer than 5 sentences
- Sentences with fewer than 3 words
- References to Javascript
- Placeholder “Lorem ipsum” text
- Obsceneties

How much data is 750GB?

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

Play with the data: `https://c4-search.apps.allenai.org/`

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

Pre-training Data: Experiment

Takeaway:

- Clean and compact data is better than large, but noisy data.
- Pre-training on in-domain data helps.

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**

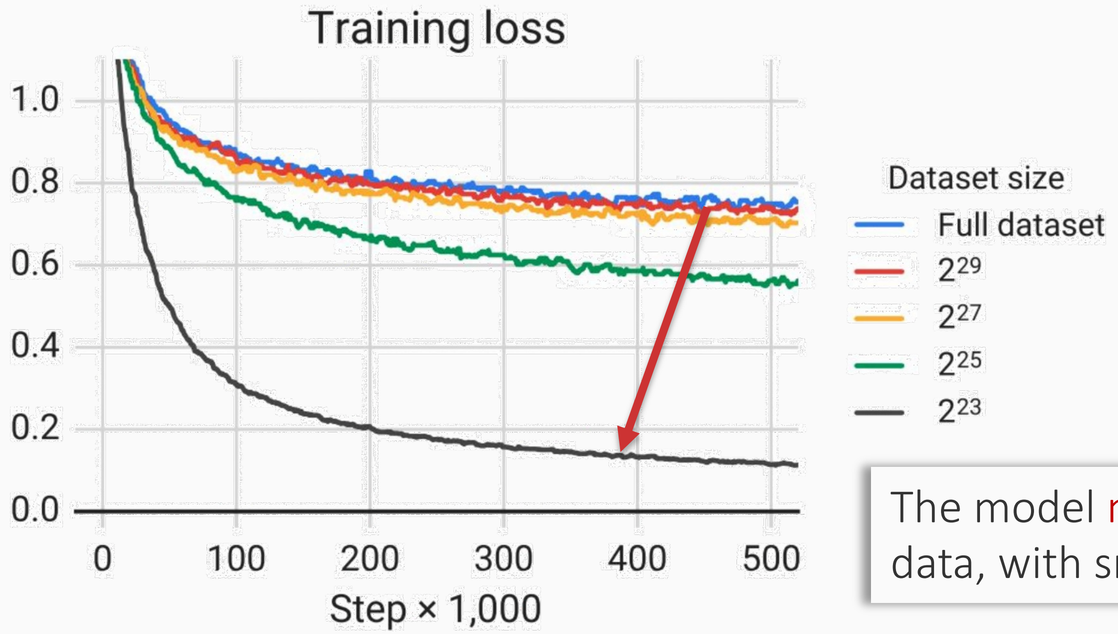
What happens if there are duplicates in the data?

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

Performance degrades as the information content shrinks

What happens if there are duplicates in the data?

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



The model memorizes the pre-training data, with smaller/repeated datasets

Key findings (recap)

Model Architectures	Encoder-decoder models outperform "decoder-only" language models
Pre-training Objectives	Fill-in-the-blank-style denoising objectives are most effective. Computational cost is a crucial factor
Unlabeled Datasets	Training on in-domain data is beneficial, but pre-training on smaller datasets can lead to overfitting
Training Strategies	Multitask learning is competitive with pre-train-then-fine-tune, but task frequency needs careful consideration
Scale	Comparison of scaling up model size, training time, and ensembled models for optimal use of fixed compute power

We have already seen some scaling results

The T5 model family

Name	d_{model}	d_{ff}	d_{kv}	Attention Heads	Encoder Layers	Decoder Layers	Size
Small	512	2,048	64	8	6	6	~60M
Base	768	3,072	64	12	12	12	~220M
Large	1,024	4,096	64	16	24	24	~770M
3B	1,024	16,384	128	32	24	24	~2.8B
11B	1,024	65,536	128	128	24	24	~11B

A sampling of model performance

Model	GLUE Average	SST-2 Accuracy	MRPC F1	STS-B Spearman	MNLI-m Accuracy	MNLI-mm Accuracy	SQuAD F1	SuperGLUE Average	BoolQ Accuracy
Previous best	89.4	97.1	93.6	92.3	91.3	91.0	95.5	84.6	87.1
T5-Small	77.4	91.8	89.7	85.0	82.4	82.3	87.24	63.3	76.4
T5-Base	82.7	95.2	90.7	88.6	87.1	86.2	92.08	76.2	81.4
T5-Large	86.4	96.3	92.4	89.2	89.9	89.6	93.79	82.3	85.4
T5-3B	88.5	97.4	92.5	89.8	91.4	91.2	94.95	86.4	89.9
T5-11B	**90.3**	**97.5**	**92.8**	**92.8**	**92.2**	**91.9**	**96.22**	**88.9**	**91.2**

General trends
- Better than previous best results
- Larger models perform better

BART (Lewis et al. 2020)

Similar architecture as T5

- Performs competitive to RoBERTa and XLNet on discriminative/classification tasks
- Outperformed existing methods on generative tasks (question answering, and summarization)
- Improved results on machine translation with fine-tuning on target language

BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension

Mike Lewis*, Yinhan Liu*, Naman Goyal*, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, Luke Zettlemoyer
Facebook AI
{mikelewis,yinhanliu,naman}@fb.com

Summary

- T5 and BART: Encoder decoder models
- General idea: Convert all NLP tasks into a format that the encoder-decoder can accept
  - Pretrain on large data
  - Fine-tune on many different tasks together
- Easy to use today using HuggingFace