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

Colin Raffel • Noam Shazeer • Adam Roberts • Katherine Lee • Sharan Narang • Michael Matena • Yanqi Zhou • Wei Li • Peter J. Liu

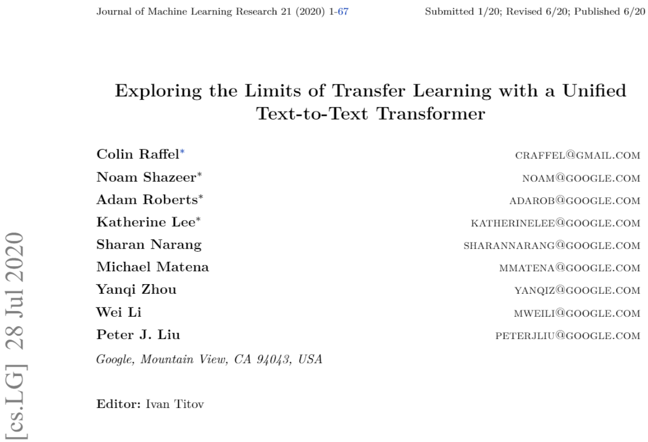
Nikita Nikolaev

Novosibirsk State University

04/27/2021

Outline

Foreword


Journal of Machine Learning Research 21 (2020) 1-67 Submitted 1/20; Revised 6/20; Published 6/20

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

Figure 1: https://arxiv.org/abs/1910.10683v3

Text-to-Text Transfer Transformer

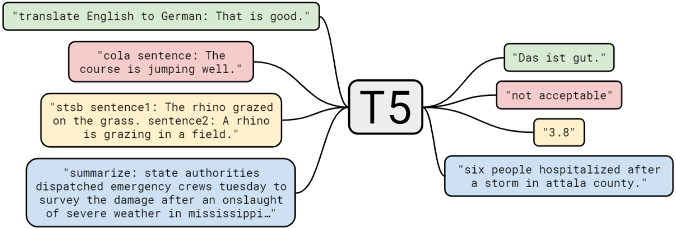


Figure 2: A diagram of text-to-text framework. Every task authors consider — including translation, question answering, and classification — is cast as feeding our model text as input and training it to generate some target text. This allows to use the same model, loss function, hyperparameters, etc. across diverse set of tasks. It also provides a standard test bed for the methods included in this empirical survey."T5" refers to the proposed model, which authors dub the "Text-to-Text Transfer Transformer".

Model - Transformer-based Architecture

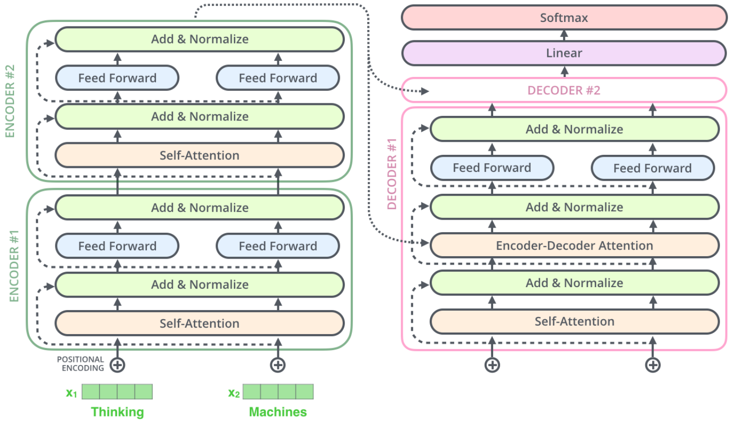


Figure 3: The Transformer - model architecture. From 'Attention Is All You Need' by Vaswani et al.

Data - Colossal Clean Crawled Corpus

Common Crawl - a publicly-available web archive - the basis for C4 dataset.
Heuristics for cleaning up Common Crawl's web extracted text:

- Authors only retained lines that ended in a terminal punctuation mark (i.e. a period, exclamation mark, question mark, or end quotation mark).
- Authors discarded any page with fewer than 5 sentences and only retained lines that contained at least 3 words.
- Authors removed any page that contained any word on the "List of Dirty, Naughty, Obscene or Otherwise Bad Words".
- Many of the scraped pages contained warnings stating that Javascript should be enabled so Authors removed any line with the word Javascript.
- Some pages had placeholder "lorem ipsum" text; Authors removed any page where the phrase "lorem ipsum" appeared.
- Some pages inadvertently contained code. Since the curly bracket appears in many programming languages (such as Javascript, widely used on the web) but not in natural text, Authors removed any pages that contained a curly bracket.
- To deduplicate the data set, Authors discarded all but one of any three-sentence span occurring more than once in the data set.

Downstream Tasks

The goal in this paper is to measure general language learning abilities. As such, Authors study downstream performance on a diverse set of benchmarks, including:

- GLUE and SuperGLUE text classification meta-benchmarks
- CNN/Daily Mail abstractive summarization
- SQuAD question answering
- WMT English to German, French, and Romanian translation

GLUE and SuperGLUE each comprise a collection of text classification tasks meant to test general language understanding abilities:

- Sentence acceptability judgment (CoLA)
- Sentiment analysis (SST-2)
- Paraphrasing/sentence similarity (MRPC, STS-B, QQP)
- Natural language inference (MNLI, QNLI, RTE, CB)
- Coreference resolution (WNLI and WSC)
- Sentence completion (COPA)
- Word sense disambiguation (WIC)
- Question answering (MultiRC, ReCoRD, BoolQ)

Input and Output Format of Data

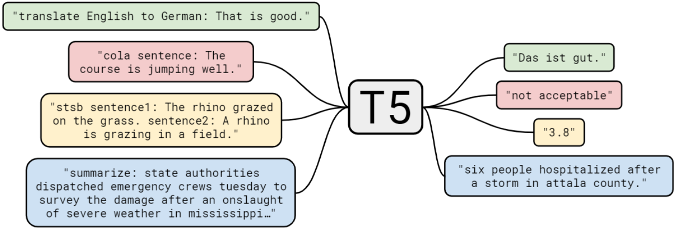


Figure 4: A diagram of text-to-text framework. Every task authors consider — including translation, question answering, and classification — is cast as feeding our model text as input and training it to generate some target text. This allows to use the same model, loss function, hyperparameters, etc. across diverse set of tasks. It also provides a standard test bed for the methods included in this empirical survey."T5" refers to the proposed model, which authors dub the "Text-to-Text Transfer Transformer".

Baseline

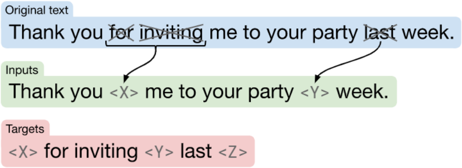


Figure 5: Schematic of the objective authors use in the baseline model. In this example, authors process the sentence "Thank you for inviting me to your party last week." The words "for", "inviting" and "last" (marked with an $\times$) are randomly chosen for corruption. Each consecutive span of corrupted tokens is replaced by a sentinel token (shown as <X> and <Y>) that is unique over the example. Since "for" and "inviting" occur consecutively, they are replaced by a single sentinel <X>. The output sequence then consists of the dropped-out spans, delimited by the sentinel tokens used to replace them in the input plus a final sentinel token <Z>.

Baseline - Results

	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

Figure 6: Average and standard deviation of scores achieved by the baseline model and training procedure. For comparison, authors also report performance when training on each task from scratch (i.e. without any pre-training) for the same number of steps used to fine-tune the baseline mode.

Architectures, Model Structures and Its Comparison

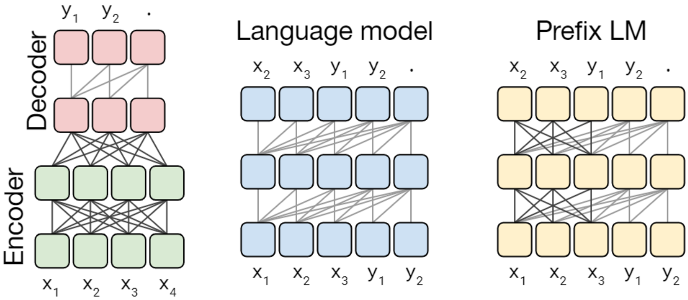


Figure 7: Schematics of the Transformer architecture variants authors consider

Architectures, Model Structures and Its Comparison

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

Figure 8: Performance of the different architectural variants. Authors use P to refer to the number of parameters in a 12-layer base Transformer layer stack and M to refer to the FLOPs required to process a sequence using the encoder-decoder model. Authors evaluate each architectural variant using a denoising objective and an autoregressive objective (as is commonly used to train language models).

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)*
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
Random spans	Thank you <X> to <Y> week .	<X> for inviting me <Y> your party last <Z>

Figure 9: Examples of inputs and targets produced by some of the unsupervised objectives authors consider.

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**

Figure 10: Comparison of variants of the BERT-style pre-training objective. In the first two variants, the model is trained to reconstruct the original uncorrupted text segment. In the latter two, the model only predicts the sequence of corrupted tokens.

Different Unlabeled 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	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**

Figure 11: Performance resulting from pre-training on different data sets. The first four variants are based on the new C4 data set.

Pre-training Data Set Size

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

Figure 12: Measuring the effect of repeating data during pre-training. In these experiments, authors only use the first N tokens from C4 (with varying values of N shown in the first column) but still pre-train over 2^{35} tokens. This results in the data set being repeated over the course of pre-training (with the number of repeats for each experiment shown in the second column), which may result in memorization.

Fine-tuning Methods

Fine-tuning method	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ All parameters	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
Adapter layers, $d = 32$	80.52	15.08	79.32	60.40	13.84	17.88	15.54
Adapter layers, $d = 128$	81.51	16.62	79.47	63.03	19.83	27.50	22.63
Adapter layers, $d = 512$	81.54	17.78	79.18	64.30	23.45	33.98	25.81
Adapter layers, $d = 2048$	81.51	16.62	79.47	63.03	19.83	27.50	22.63
Gradual unfreezing	82.50	18.95	79.17	**70.79**	26.71	39.02	26.93

Figure 13: Comparison of different alternative fine-tuning methods that only update a subset of the model's parameters. For adapter layers, *d* refers to the inner dimensionality of the adapters.

Multi-task Learning

Mixing strategy	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Baseline (pre-train/fine-tune)	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
Equal	76.13	19.02	76.51	63.37	23.89	34.31	26.78
Examples-proportional, $K = 2^{16}$	80.45	19.04	77.25	69.95	24.35	34.99	27.10
Examples-proportional, $K = 2^{17}$	81.56	19.12	77.00	67.91	24.36	35.00	27.25
Examples-proportional, $K = 2^{18}$	81.67	19.07	78.17	67.94	24.57	35.19	27.39
Examples-proportional, $K = 2^{19}$	81.42	**19.24**	79.78	67.30	25.21	36.30	**27.76**
Examples-proportional, $K = 2^{20}$	80.80	**19.24**	**80.36**	67.38	25.66	36.93	**27.68**
Examples-proportional, $K = 2^{21}$	79.83	18.79	79.50	65.10	25.82	37.22	27.13
Temperature-scaled, $T = 2$	81.90	**19.28**	79.42	69.92	25.42	36.72	27.20
Temperature-scaled, $T = 4$	80.56	**19.22**	77.99	69.54	25.04	35.82	27.45
Temperature-scaled, $T = 8$	77.21	19.10	77.14	66.07	24.55	35.35	27.17

Figure 14: Comparison of multi-task training using different mixing strategies. Examples-proportional mixing refers to sampling examples from each data set according to the total size of each data set, with an artificial limit (K) on the maximum data set size. Temperature-scaled mixing re-scales the sampling rates by a temperature T. For temperature-scaled mixing, we use an artificial data set size limit of $K = 2^{21}$.

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**

Figure 15: Comparison of unsupervised pre-training, multi-task learning, and various forms of multi-task pre-training.

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

Figure 16: Comparison of different methods of scaling up our baseline model. All methods except ensembling fine-tuned models use 4× the computation as the baseline. "Size" refers to the number of parameters in the model and "training time" refers to the number of steps used for both pre-training and fine-tuning.

Pushing the limits

Up to 11 billion model parameters
Over 1 trillion tokens for training

Thank You for Your Attention!