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

T5 model

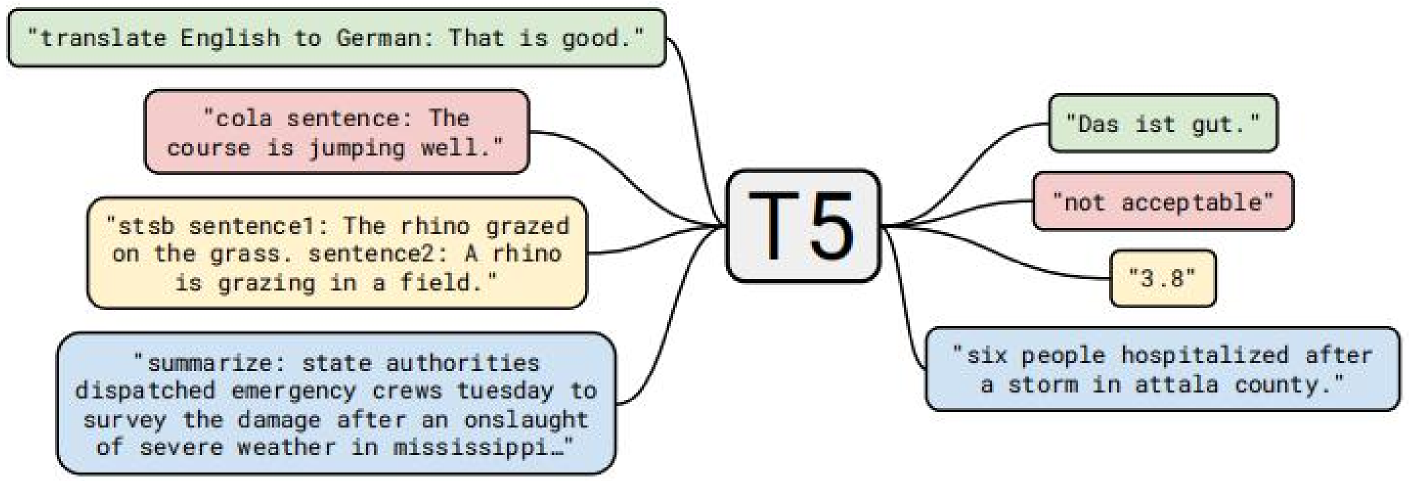

"translate English to German: That is good."
"cola sentence: The course is jumping well."
"stsb sentence1: The rhino grazed on the grass. sentence2: A rhino is grazing in a field."
"summarize: state authorities dispatched emergency crews tuesday to survey the damage after an onslaught of severe weather in mississippi…"
T5
"Das ist gut."
"not acceptable"
"3.8"
"six people hospitalized after a storm in attala county."

Figure 1: A diagram of our text-to-text framework. Every task we 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 us to use the same model, loss function, hyperparameters, etc. across our diverse set of tasks. It also provides a standard testbed for the methods included in our empirical survey. "T5" refers to our model, which we dub the "**T**ext-**t**o-**T**ext **T**ransfer **T**ransformer".

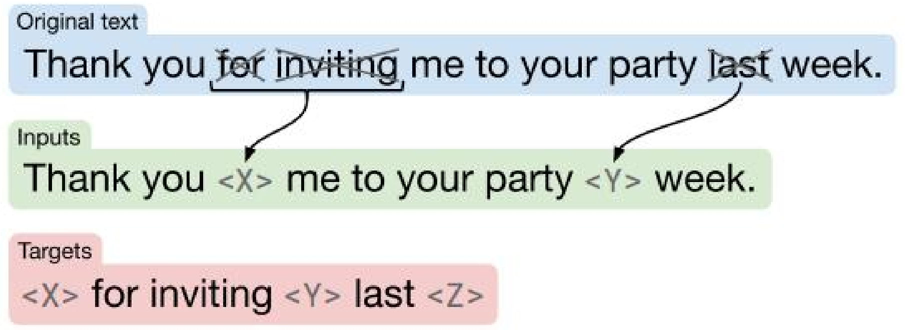
T5 的基本思想是将每个 NLP 问题都视为"**text-to-text**"问题，即将文本作为输入并生成新的文本作为输出，这允许将相同的模型、目标、训练步骤和解码过程，直接应用于每个任务。

The Colossal Clean Crawled Corpus **745G**

Much of the previous work on transfer learning for NLP makes use of large unlabeled data sets for unsupervised learning. In this paper, we are interested in measuring the effect of the quality, characteristics, and size of this unlabeled data. To generate data sets that satisfy our needs, we leverage Common Crawl as a source of text scraped from the web. Common

Input and Output Format

In order to train a single model on the diverse set of tasks described above, we cast all of the tasks we consider into a "text-to-text" format—that is, a task where the model is fed some text for context or conditioning and is then asked to produce some output text. This framework provides a consistent training objective both for pre-training and fine-tuning. Specifically, the model is trained with a maximum likelihood objective (using "teacher forcing" (Williams and Zipser, 1989)) regardless of the task. To specify which task the model should perform, we add a task-specific (text) prefix to the original input sequence before feeding it to the model.



Original text
Thank you ~~for~~ ~~inviting~~ me to your party ~~last~~ week.
Inputs
Thank you <X> me to your party <Y> week.
Targets
<X> for inviting <Y> last <Z>

Figure 2: Schematic of the objective we use in our baseline model. In this example, we 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>`.

	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

Table 1: Average and standard deviation of scores achieved by our baseline model and training procedure. For comparison, we 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 model. All scores in this table (and every table in our paper except Table 14) are reported on the validation sets of each data set.

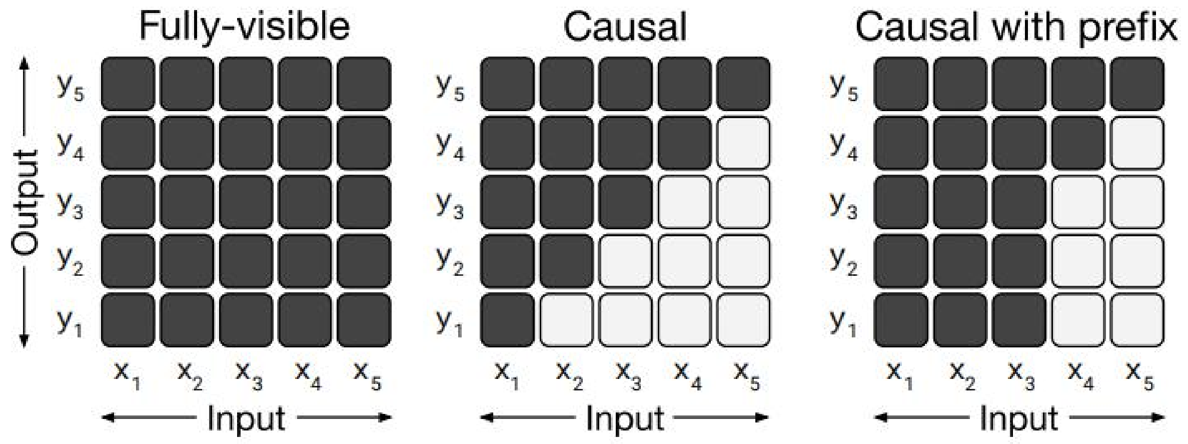
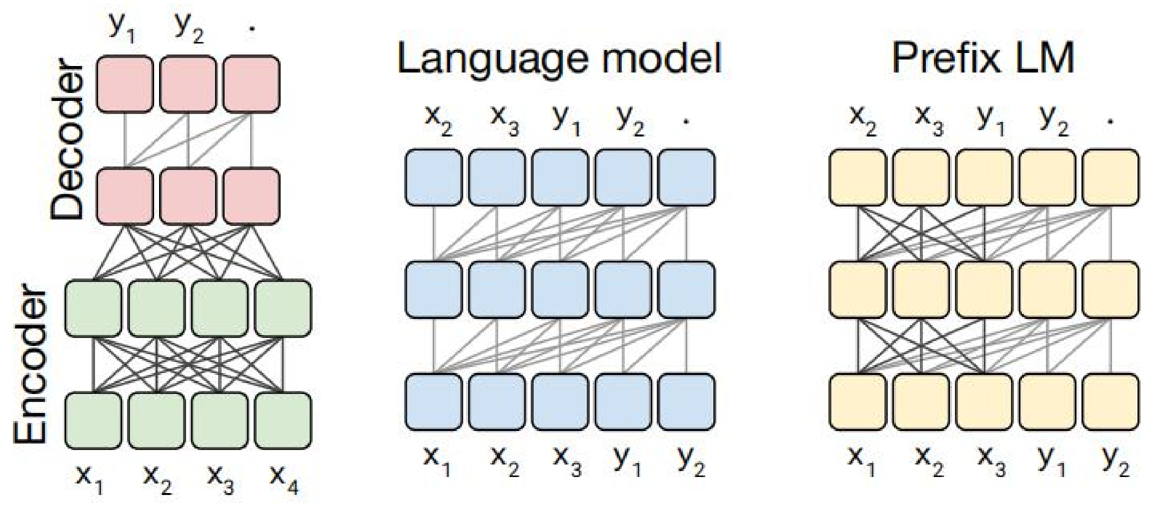


Figure 3: Matrices representing different attention mask patterns. The input and output of the self-attention mechanism are denoted x and y respectively. A dark cell at row i and column j indicates that the self-attention mechanism is allowed to attend to input element j at output timestep i. A light cell indicates that the self-attention mechanism is *not* allowed to attend to the corresponding i and j combination. Left: A fully-visible mask allows the self-attention mechanism to attend to the full input at every output timestep. Middle: A causal mask prevents the ith output element from depending on any input elements from "the future". Right: Causal masking with a prefix allows the self-attention mechanism to use fully-visible masking on a portion of the input sequence.

左：完全可见的掩码。输出的每个时间步会注意全部输入

中：因果掩码。防止第 i 个输出元素依赖于"未来"的任何输入元素

右：带前缀的因果掩码。使自我注意机制可以在输入序列的一部分上使用完全可见的掩码



Decoder
Encoder
Language model
Prefix LM
y_1 y_2 .
x_1 x_2 x_3 x_4
x_2 x_3 y_1 y_2 .
x_1 x_2 x_3 y_1 y_2
x_2 x_3 y_1 y_2 .
x_1 x_2 x_3 y_1 y_2

Figure 4: Schematics of the Transformer architecture variants we consider. In this diagram, blocks represent elements of a sequence and lines represent attention visibility. Different colored groups of blocks indicate different Transformer layer stacks. Dark grey lines correspond to fully-visible masking and light grey lines correspond to causal masking. We use "." to denote a special end-of-sequence token that represents the end of a prediction. The input and output sequences are represented as x and y respectively. Left: A standard encoder-decoder architecture uses fully-visible masking in the encoder and the encoder-decoder attention, with causal masking in the decoder. Middle: A language model consists of a single Transformer layer stack and is fed the concatenation of the input and target, using a causal mask throughout. Right: Adding a prefix to a language model corresponds to allowing fully-visible masking over the input.

左：标准的编码器-解码器体系结构，
在注意力中使用完全可见的掩码，
在解码器中使用因果掩码

中：使用因果掩码

右：在语言模型中添加前缀并使用完全可见的掩码

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

Table 2: Performance of the different architectural variants described in Section 3.2.2. We 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. We evaluate each architectural variant using a denoising objective (described in Section 3.1.4) and an autoregressive objective (as is commonly used to train language models).

Augmented Natural Language for Generative Sequence Labeling

Ben Athiwaratkun
AWS AI
benathi@amazon.com

Cicero Nogueira dos Santos
AWS AI
cicnog@amazon.com

Jason Krone
AWS AI
kronej@amazon.com

Bing Xiang
AWS AI
bxiang@amazon.com

Summary

- Proposed a generative framework for joint sequence labeling and sentence-level classification.
- Proposed an effective new output format to perform joint sequence labeling and sentence classification.

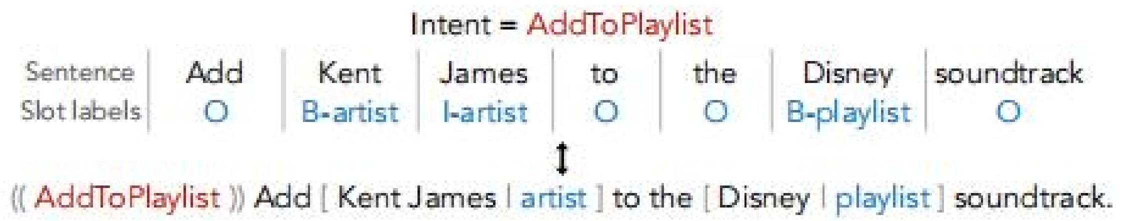


Figure 1: The conversion between the canonical BIO tagging format and our augmented natural language format.

We propose a new formulation for this generation task such that, given the input sequence ℓ, our method generates output o in **augmented natural language**. The augmented output o repeats the original input sequence ℓ with additional markers that indicate the token-spans and their associated labels. More specifically, we use the format $[\ \ell_j, \ldots, \ell_{j+t} \mid\ L\ ]$ to indicate that the token sequence $\ell_j, \ldots, \ell_{j+t}$ is labeled as L.

Joint Sequence Classification and Labeling

Our sequence to sequence approach also supports joint sentence classification and sequence labeling by incorporating the sentence-level label in the augmented natural language format. In practice, we use the pattern ((sentence-level label)) in the beginning of the generated sentence, as shown in Fig. 1. The use of double parentheses is to prevent confusion with a single parenthesis that can occur in the original word sequence ℓ.

Natural Labels

We perform label mapping in order to match the labels to its natural descriptions and use the natural labels in the augmented natural language output. Our motivation is as follows: (1) Pre-trained conditional generation models which we adapt on have richer semantics embedded in natural words, rather than dataset-specific label names. For instance, "country city state" contains more semantic information compared to "GPE", which is an original label in named entity recognition tasks. Using natural labels should allow the model to learn the association between word tokens and labels more efficiently, without requiring many examples. (2) Label knowledge can be shared among different tasks. For instance, after learning how to label names as "person", given a new task in another domain which requires labeling "artist", the model can more easily associate names with "artist" due to the proximity of "person" and "artist" in embeddings. This is not the case if the concept of "person" was learned with other uninformative words.

Ontonotes slot types	Natural-word label	Descriptions
CARDINAL	cardinal	Numerals that do not fall under another type
DATE	date	Absolute or relative dates or periods
EVENT	event	Named hurricanes, battles, wars, sports events, etc.
FAC	facility	Buildings, airports, highways, bridges, etc.
GPE	country city state	countries, cities, states
LANGUAGE	language	Any named language
LAW	law	Named documents made into laws
LOC	location	Non-GPE locations, mountain ranges, bodies of water
MONEY	money	Monetary values, including unit
NORP	nationality religious political group	Nationalities or religious or political groups
ORDINAL	ordinal	”first”, ”second”
ORG	organization	Companies, agencies, institutions, etc.
PERCENT	percent	Percentage (including “%”)
PERSON	person	People, including fictional
PRODUCT	product	Vehicles, weapons, foods, etc. (Nor services)
QUANTITY	quantity	Measurements, as of weight or distance
TIME	time	Times smaller than a day
WORK_OF_ART	work of art	Titiles of books, songs, etc.

Table 9: Label mapping for Ontonotes. The descriptions are obtained from Weischedel et al. (2012)

CONLL-2003 slot types	Natural-word label
LOC	location
MISC	miscellaneous
ORG	organization
PER	person

Adapting the **pre-trained T5** with the sequence to sequence framework

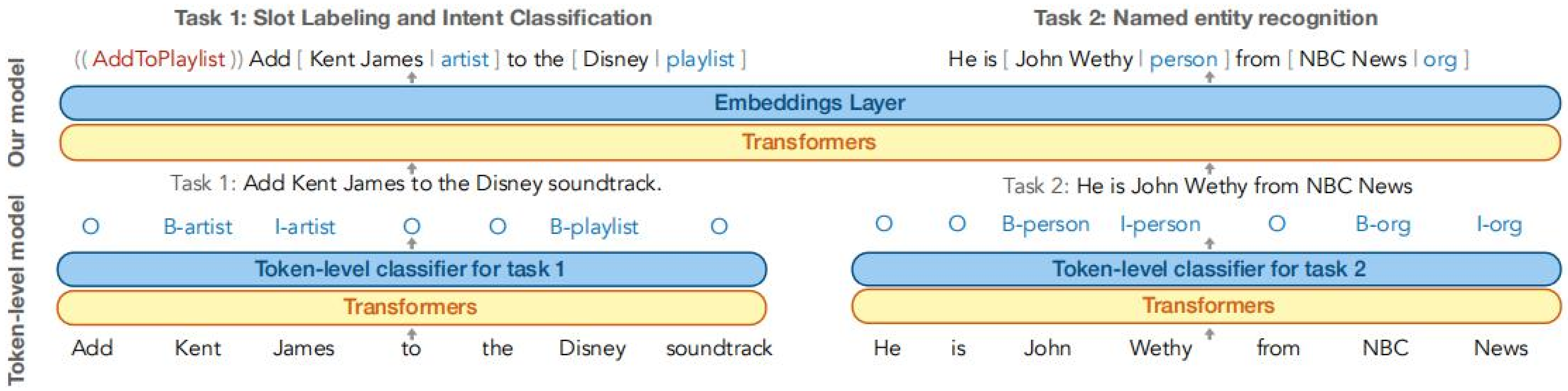

Task 1: Slot Labeling and Intent Classification
Task 2: Named entity recognition
Our model
((AddToPlaylist)) Add [Kent James | artist] to the [Disney | playlist]
He is [John Wethy | person] from [NBC News | org]
Embeddings Layer
Transformers
Task 1: Add Kent James to the Disney soundtrack.
Task 2: He is John Wethy from NBC News
Token-level model
O B-artist I-artist O O B-playlist O
O O B-person I-person O B-org I-org
Token-level classifier for task 1
Token-level classifier for task 2
Transformers
Transformers
Add Kent James to the Disney soundtrack
He is John Wethy from NBC News

Figure 2: Comparison between our generative-style sequence labeling model (top) and the conventional token-level classification model (bottom).

Experiments

- SNIPS(Coucke et al., 2018)---7 intents and 39 distinct types of slots
- ATIS (Hemphill et al.,1990)
- Ontonotes(Pradhan et al., 2013)
- CoNLL-2003 (Sang and Meulder, 2003)

	Task & Dataset	Intent Clas.		Slot Labeling			
		SNIPS	ATIS	SNIPS	ATIS	CoNLL	Onto
SL/IC	Bi-Model (Wang et al., 2018)		**98.99**		**96.89**		
	Joint BERT (Chen et al., 2019)	98.60	97.50	**97.00**	96.10		
	ELMO+BiLSTM (Siddhant et al., 2019)	**99.29**	97.42	93.90	95.62		
NER	Cloze-CNN (Baevski et al., 2019)					**93.50**	
	BERT-MRC (Li et al., 2019a)					93.04	91.11
	BERT-MRC + DSC (Li et al., 2019b)					93.33	**92.07**
	BERT Base (Devlin et al., 2019)					92.40	88.95
	Ours: Individual	99.00	96.86	**97.43**	96.13	90.70	**90.24**
	Ours: SNIPS+ATIS	**99.29**	**97.20**	97.21	95.83		
	Ours: CoNLL+Ontonotes					**91.48**	89.52
	Ours: SNIPS+ATIS+CoNLL+Ontonotes	99.14	97.08	96.82	**96.65**	**91.48**	89.67