Pseudo Lab NLP Study

[T5] Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer (Raffel et al., 2019. 10.)

https://doi.org/10.48550/arXiv.1910.10683

250417 Bowon Kwon

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.

T5 논문 목차

0. Evolution of NLP Models (1)

- **RNN** (1980s ~): 순차적 데이터 처리, e.g., Elman (1990) 은닉층 통해 이전 문맥 정보 유지
  - **LSTM** (Hochreiter & Schmidhuber, 1997. 11.): 장기 의존성 문제 해결 위해 입력/출력/망각 게이트를 포함한 메모리 셀 구조 도입하여 정보의 선택적 유지 및 제거 가능
  - **BiRNN** (Bidirectional RNN, Schuster & Paliwal, 1997. 11.): 입력 시퀀스를 양방향으로 처리하여 앞뒤 문맥 모두 반영 ⇢ ELMo (Peters et al., 2018. 2.)는 BiLSTM (GPT-1 이전)
  - **GRU** (Cho et al., 2014. 6.): LSTM을 단순화한 게이트 구조
- **Seq2Seq** (Sutskever, Vinyals & Le, 2014. 9.): LSTM Encoder + LSTM Decoder 구조로, 인코더가 입력 시퀀스를 고정 길이 벡터로 압축한 후 디코더가 출력 시퀀스 생성
- **Seq2Seq + Attention** (Bahdanau, Cho & Bengio, 2014. 9.): Decoder가 Encoder의 모든 hidden state에 가중치 부여하여 관련성 높은 정보에 집중하는 Attention 메커니즘 도입
- **Transformer** (Vaswani et al., 2017. 6.): RNN 없는 Self-Attention 기반의 Multi-Head Encoder + Decoder 구조로, 병렬 처리 효율성 및 문맥 포착 능력 대폭 향상

0. Evolution of NLP Models (2)

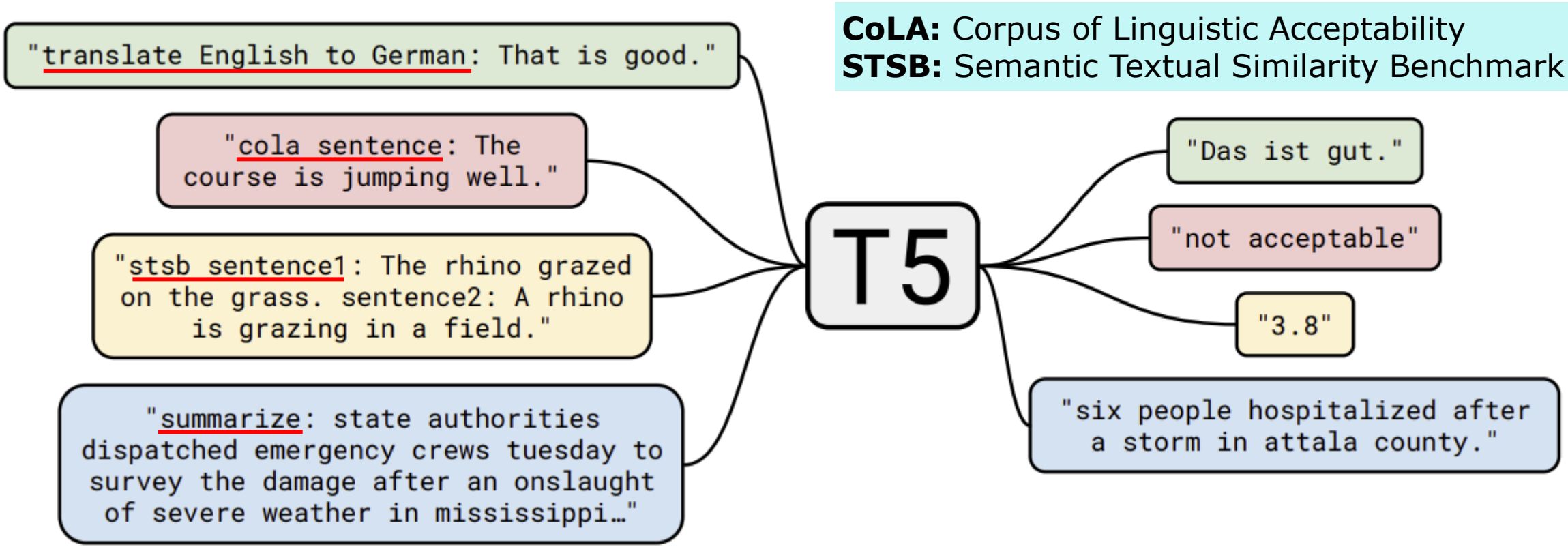
- Transformer **Decoder**-only
  - **GPT-1** (Radford et al., 2018. 6.): Pre-Training + Fine-Tuning, 117M parameters
  - **GPT-2** (Radford et al., 2019. 2.): 1.5B 모델과 대규모 데이터로 Fine-Tuning 없이도 Zero-Shot 능력 확인, Cf. T5 논문은 2019. 10. arxiv 게시
  - **XLNet** (Yang et al., 2019. 6.): Transformer-XL (Extra Long, Dai et al., 2019. 1.) 의 AutoRegressive Pre-Training에 Permutation 적용하여 BERT처럼 양방향 문맥 반영
  - **GPT-3** (Brown et al., 2020. 5.): 175B 초대형 모델로 Few-Shot 능력 검증
  - **InstructGPT** (Ouyang et al., 2022. 5.): RLHF ⇢ ChatGPT (2022. 11.)
- Transformer **Encoder**-only
  - **BERT** (Devlin et al., 2018. 7.): Pre + Fine, Bidirectional Self-Attention
  - **MT-DNN** (Multi-Task DNN, Liu et al., 2019. 1.): BERT 기반 Multi-Task Learning
  - **RoBERTa** (Liu et al., 2019. 7.): BERT의 Pre-Training 개선하여 성능 향상
  - **ALBERT** (Lan et al., 2019. 9.): Cross-Layer Parameter Sharing, Factorized Embedding Parameterization으로 BERT 경량화

0. Evolution of NLP Models (3)

- Transformer **Encoder** (BERT의 **A**uto**E**ncoder) **+** **Decoder** (GPT의 **A**uto**R**egressive)
  - **MASS** (Masked Seq2Seq Pre-Training, Song et al., 2019. 5.): Encoder에서 입력 일부를 Masking하고 Decoder가 해당 부분을 복원하도록 학습 (Cf. MASS가 XLNet보다 먼저 나왔다고 정리하는 경우가 있으나, MASS는 늦어도 2019. 6. 9.~15. 열린 ICML 2019에서 발표되었고, XLNet의 arxiv 게시일은 2019. 6. 19.임, 또한 XLNet을 AE+AR로 분류하는 경우가 있고 이전 자료에서 이를 따랐으나, 수정본에서 Decoder-only로 이동하였음)
  - **T5** (Raffel et al., 2019. 10. 23.): NLP Task를 Text-to-Text 형태로 통일
  - **BART** (Lewis et al., 2019. 10. 29.): 다양한 Noising (Token Masking & Deletion, Sentence Permutation…)으로 입력 전체를 망가뜨렸다가 전체 문장 복원하도록 학습
- **Transformer** (2017. 6.) [ELMo (2018. 2.)] → GPT-1 (2018. 6.) → BERT (2018. 7.) → MT-DNN (2019. 1.) → GPT-2 (2019. 2.) → MASS (2019. 5.) → XLNet (2019. 6.) → RoBERTa (2019. 7.) → ALBERT (2019. 9.) → **T5** (2019. 10.) → BART (2019. 10.) → GPT-3 (2020. 5.) → InstructGPT (2022. 5.) → **ChatGPT** (2022. 11.)

1. Introduction

- Google 연구자들의 실험 논문(“[O]ur goal is not to propose new methods but instead to provide a comprehensive perspective on where the field stands.” p. 3)
- ML 모델을 저수준(철자, 단어 의미)부터 고수준(상식 추론) 포괄하는 다양한 downstream 과제에 맞추는 것을 텍스트를 “이해”할 수 있는 범용 지식 갖추게 하는 것에 빗대어 생각할 수 있음
  - 언어 지식은 직접 주입되기보다는 다른 보조 과제를 수행하는 과정에서 간접적으로 학습됨
  - 전통적으로 Word2Vec (Mikolov et al. 2013), GloVe (Pennington et al., 2014) 등 단어 임베딩을 통해 이루어졌고,
  - 풍부한 데이터로 전체 모델을 사전 훈련하여 범용 능력을 갖추게 한 뒤 downstream 과제에 전이하는 pre-train-then-fine-tune approach가 보편화되고 있음(사전학습 방식은 Computer Vision에서는 대규모 labeled data 지도학습, NLP에서는 Common Crawl 등 대규모 unlabeled data 비지도 학습)
- NLP 분야에서 다양한 전이학습 연구가 이루어지고 있으나, 사전학습 목표, 데이터셋, 벤치마크, 파인튜닝 방식 등이 다양하여 통합적 비교분석 어려움
  - 모든 NLP 과제를 통합된(unified) Text-to-Text 문제(input text → output text)로 재정의한 후, 다양한 실험을 통해 현 시점에서 NLP 전이학습의 성능 한계를 탐색
  - C4 (Colossal Clean Crawled Corpus) 데이터셋, 코드, 사전학습 모델 공개



CoLA: Corpus of Linguistic Acceptability
STSB: Semantic Textual Similarity Benchmark

Task-specific (text) prefix

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".

2.1. Model

- Transformer Encoder + Decoder, 그러나 다음과 같은 차이
- **Pre-LN**: Vaswani et al.(2017)는 Layer Normalization을 각 Sublayer 뒤에 배치하는 Post-LN이었으나($LN(x + Sublayer(x))$), T5는 residual connection 전에 배치($x + Sublayer(LN(x))$)
  - Cf. GPT-2 (2019) "Layer normalization was moved to the input of each sub-block, similar to a pre-activation residual network (He et al., 2016)"
  - Post-LN은 네트워크 깊어질수록 그레디언트 흐름이 불안정해지는 경향이 있어 학습률 warm-up 필요
  - Xiong et al. (2020) 다양한 실험 통해 Pre-LN 구조가 학습 안정성 면에서 우수하고 학습 속도가 빠름을 보임
- **Bias 없이** rescale만 하는 간단한 LayerNorm 사용(정규화된 값의 왜곡 방지 vs. 유연한 표현력?)

$$\text{LayerNorm}(x) = \frac{x - \mu}{\sqrt{\sigma^2 + \epsilon}} \cdot \gamma + \beta$$

- **Relative Position Embeddings:** Sinusoidal (Vaswani et al.(2017)), Learned (BERT, GPT-1 등) 대신 Key와 Query 상대적 거리 기준 PE(같은 단어가 다른 위치에 있어도 패턴 학습 가능)

2.2. The Colossal Clean Crawled Corpus

- **Common Crawl**
  - 매달 20TB 분량 HTML에서 텍스트 추출한 공개 웹 아카이브
  - 그러나 대부분 비자연어 텍스트이고 불필요하거나 유해한 부분도 있어 정제 필요
- **정제 기준(heuristics for cleaning up)**
  - 종결하는 문장 부호로 끝나는 문장만 사용
  - 3문장 이상 페이지, 5단어 이상 포함된 문장만 유지
  - “List of Dirty, Naughty, Obscene or Otherwise Bad Words”, “Javascript” 포함 행, lorem ipsum (자리 채우기) text, “{” 포함 페이지(코드로 간주), Wikipedia의 인용 부호([1], [citation needed] 등), “terms of use” 등 boilerplate policy 제거
  - 3문장 span이 반복하여 등장할 경우 제거
  - langdetect 이용하여 영어 텍스트만 유지($p \geq 0.99$)
- **C4:** 2019년 4월 Common Crawl 데이터에 위 기준 따른 정제 거쳐 750GB 영어 텍스트 확보

Dodge et al. (2021)

Figure 2: Number of tokens from the 25 most represented top-level domains (left) and websites (right) in C4.EN.

2.3. Downstream Tasks

- “Our goal in this paper is to measure general language learning abilities.” p. 7
- **GLUE, SuperGLUE**
  - 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, WSC)
  - Sentence completion (COPA)
  - Word sense disambiguation (WIC)
  - Question answering (MultiRC, ReCoRD, BoolQ)
- **CNN/DailyMail** abstractive summarization
- **SQuAD** question answering
- **WMT** English to German, French, Romanian translation

2.4. Input and Output Format

- 과제와 무관하게 maximum likelihood 목표로 학습, teacher forcing (정답 시퀀스를 다음 입력으로 넣어 줌, Williams & Zipser, 1989)
- 다양한 과제에 대한 Text-to-Text format 예시는 Appendix D. 참조
- 문장 유사도 판단하는 STS-B만은 1~5 사이 score를 0.2씩 21개 class로 나누어 regression

D.16 WMT English to German

Original input: "Luigi often said to me that he never wanted the brothers to end up in court," she wrote.

Processed input: translate English to German: "Luigi often said to me that he never wanted the brothers to end up in court," she wrote.

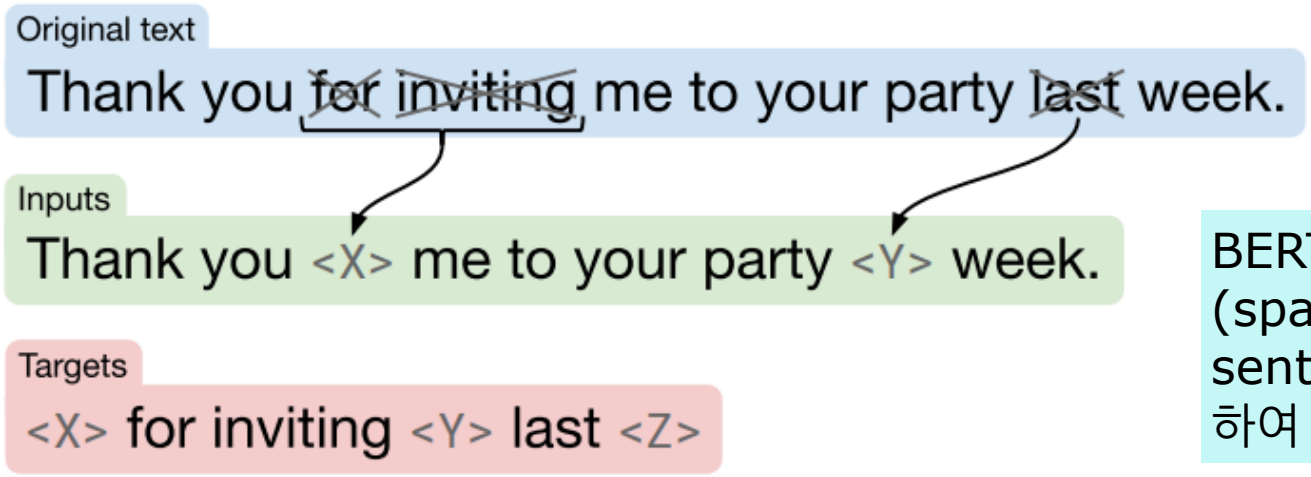
Original target: "Luigi sagte oft zu mir, dass er nie wollte, dass die Brüder vor Gericht landen", schrieb sie.

Processed target: "Luigi sagte oft zu mir, dass er nie wollte, dass die Brüder vor Gericht landen", schrieb sie.

3.1. Baseline (★)

- “Our goal for our baseline is to reflect typical, modern practice.” p. 11
- Model
  - Encoder, Decoder 각 12 blocks, ReLU, dropout 0.1, 220M parameters
- Pre-Training
  - C4로 2^{19} = 524,288 steps, maximum sequence length 512 (= 2^9), batch size 128 (= 2^7) sequences, 하나의 배치에 대략 2^{16} = 65,536 tokens 포함, 따라서 2^{35} ≈ 34B tokens로 사전 학습하는 셈이며, 이는 2.2T tokens 쓰는 BERT, RoBERTa보다 적음
  - 학습률: inverse square root ($1/\sqrt{\max(n,k)}$, n은 current training iteration, k는 number of warm-up steps = 10^4)로 조절(최초 10^4 step 동안 0.01, 이후 지수적 감소)
- Fine-Tuning
  - 2^{18} = 262,144 steps, 배치당 2^{16} tokens, 학습률 0.001 고정
- 5,000 step마다 checkpoint 저장
- “Our goal is to compare a variety of different approaches on a diverse set of tasks while keeping as many factors fixed as possible.” p. 10

3.1.4. Unsupervised Objective



BERT와 달리 연속된 토큰(span)에 하나의 고유한 sentinel token ID 부여하여 계산비용 절감

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 ×) 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>.

3.2. Architectures

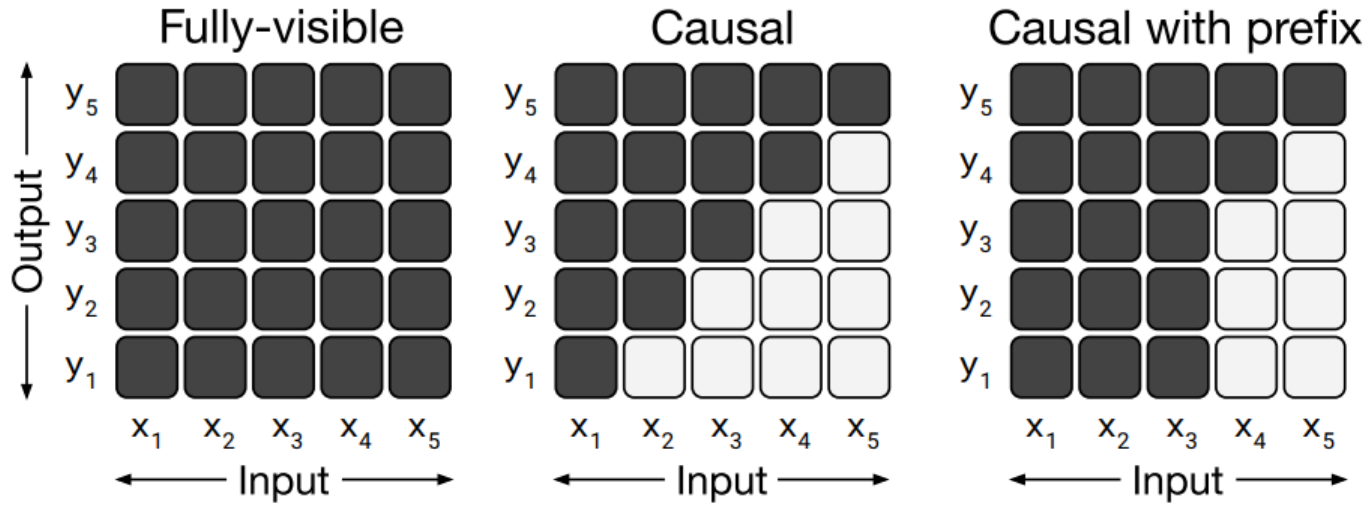


Figure 3: Matrices representing different attention mask patterns. The input and output of the self-attention mechanism are denoted x and y respectively. A dark cell at row i and column j indicates that the self-attention mechanism is allowed to attend to input element j at output timestep i. A light 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.

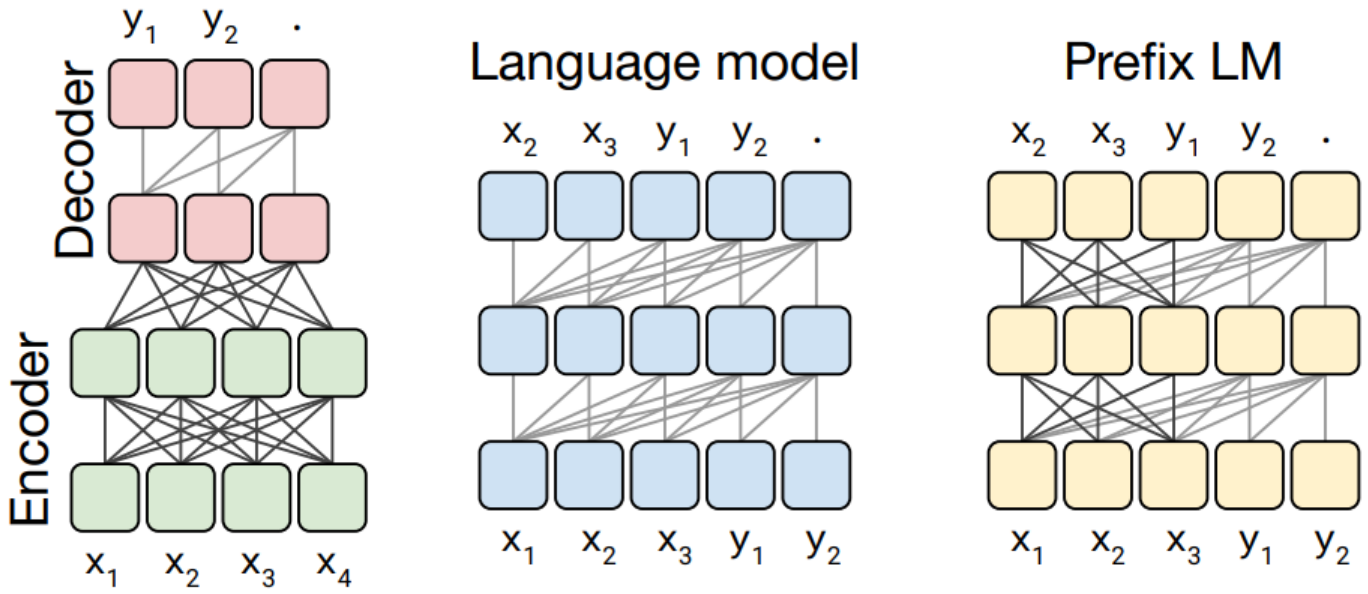


Figure 4: Schematics of the Transformer architecture variants we consider. In this diagram, blocks represent elements of a sequence and lines represent attention visibility. Different colored groups of blocks indicate different Transformer layer stacks. Dark grey lines correspond to fully-visible masking 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.

3.2. Architectures (cont'd)

3.2.2. Comparing Different Model Structures (parameters, cost)

- Encoder (L) – Decoder (L) model: 2P parameters, M FLOPs
- Parameters shared across the Encoder & Decoder: P parameters, M FLOPs
- Encoder (L/2) – Decoder (L/2) model: P parameters, M/2 FLOPs
- Decoder (L) only LM: P parameters, M FLOPs
- Decoder (L) only prefix LM: P parameters, M FLOPs

3.2.3. Objectives

- Basic Language Modeling Objective: prefix 이후 텍스트 생성
- Denoising Objective: masking된 span 생성

3.2.4. Results

- Encoder – Decoder with denoising objective performed best
- Parameter sharing으로 성능 손실 없이 경량화 가능

3.2. Architectures (cont'd)

Boldface는 within two standard deviations of the maximum (best)

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).

3.3. 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>

Baseline

Table 3: Examples of inputs and targets produced by some of the unsupervised objectives we consider applied to the input text "Thank you for inviting me to your party last week ." Note that all of our objectives process *tokenized* text. For this particular sentence, all words were mapped to a single token by our vocabulary. We write *(original text)* as a target to denote that the model is tasked with reconstructing the entire input text. <M> denotes a shared mask token and <X>, <Y>, and <Z> denote sentinel tokens that are assigned unique token IDs. The BERT-style objective (second row) includes a corruption where some tokens are replaced by a random token ID; we show this via the greyed-out word apple.

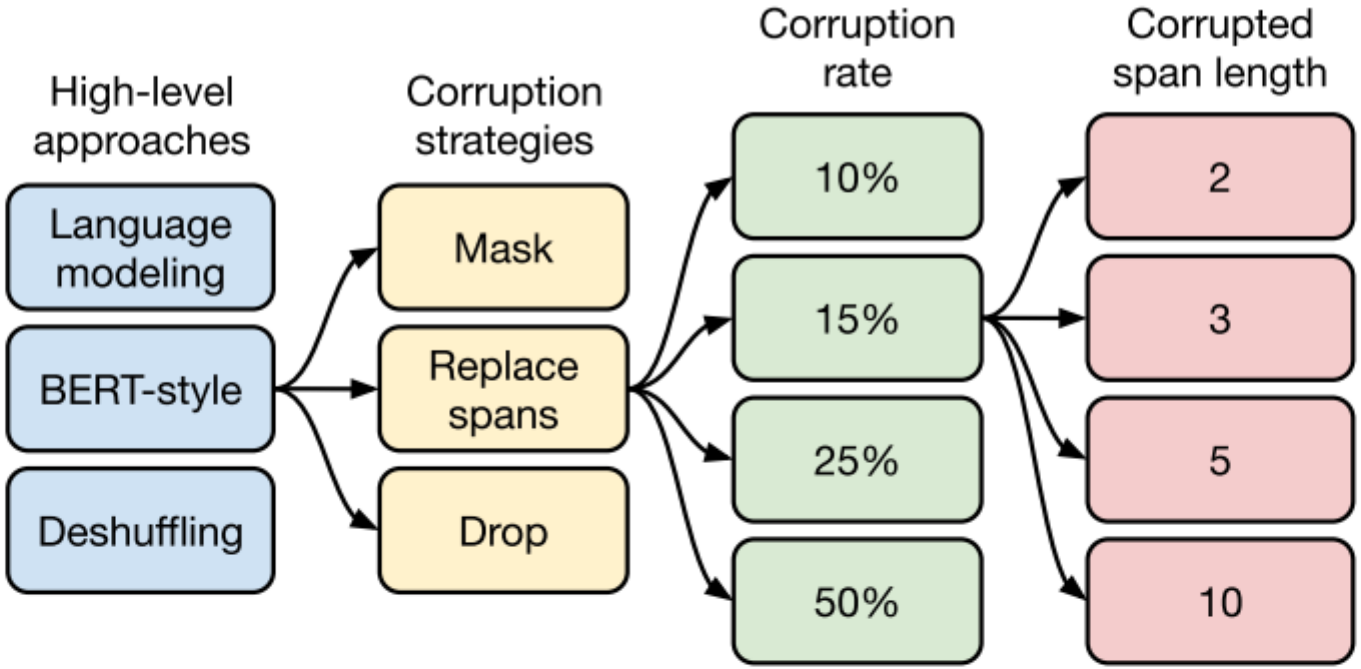


Figure 5: A flow chart of our exploration of unsupervised objectives. We first consider a few disparate approaches in Section 3.3.1 and find that a BERT-style denoising objective performs best. Then, we consider various methods for simplifying the BERT objective so that it produces shorter target sequences in Section 3.3.2. Given that replacing dropped-out spans with sentinel tokens performs well and results in short target sequences, in Section 3.3.3 we experiment with different corruption rates. Finally, we evaluate an objective that intentionally corrupts contiguous spans of tokens in Section 3.3.4.

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

Table 4: Performance of the three disparate pre-training objectives described in Section 3.3.1.

"Indeed, the motivation for the BERT objective was to outperform language model-based pre-training." p. 21

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**

Table 5: 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.

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**

Table 6: Performance of the i.i.d. corruption objective with different corruption rates.

BERT 논문도 15%

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**

Table 7: Performance of the span-corruption objective (inspired by Joshi et al. (2019)) for different average span lengths. In all cases, we corrupt 15% of the original text sequence.

3.4. Pre-Training Data Set

- 기존 연구에서는 학습 데이터 출처나 품질이 부차적으로 취급되었고, 표준 데이터셋도 부재함
- 실험 결과, 도메인 특화 데이터셋은 해당 도메인 과제에서 높은 성능
- C4와 같은 범용적이고 큰 규모의 데이터는 다양한 과제에서 균형 잡힌 성능

⟶ 가능한 한 크고 양질의 데이터 사용, 범용 + 전문 데이터 혼합

"[O]our goal is to pre-train a model that can rapidly adapt to language tasks from arbitrary domains." p. 27

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**

Table 8: Performance resulting from pre-training on different data sets. The first four variants are based on our new C4 data set.

3.4. Pre-Training Data Set (cont'd)

- 토큰 수 작게 하여 반복할수록 훈련 손실 감소하나 테스트 성능은 악화("암기" 가능성 시사)

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

Table 9: Measuring the effect of repeating data during pre-training. In these experiments, we 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 (see Figure 6).

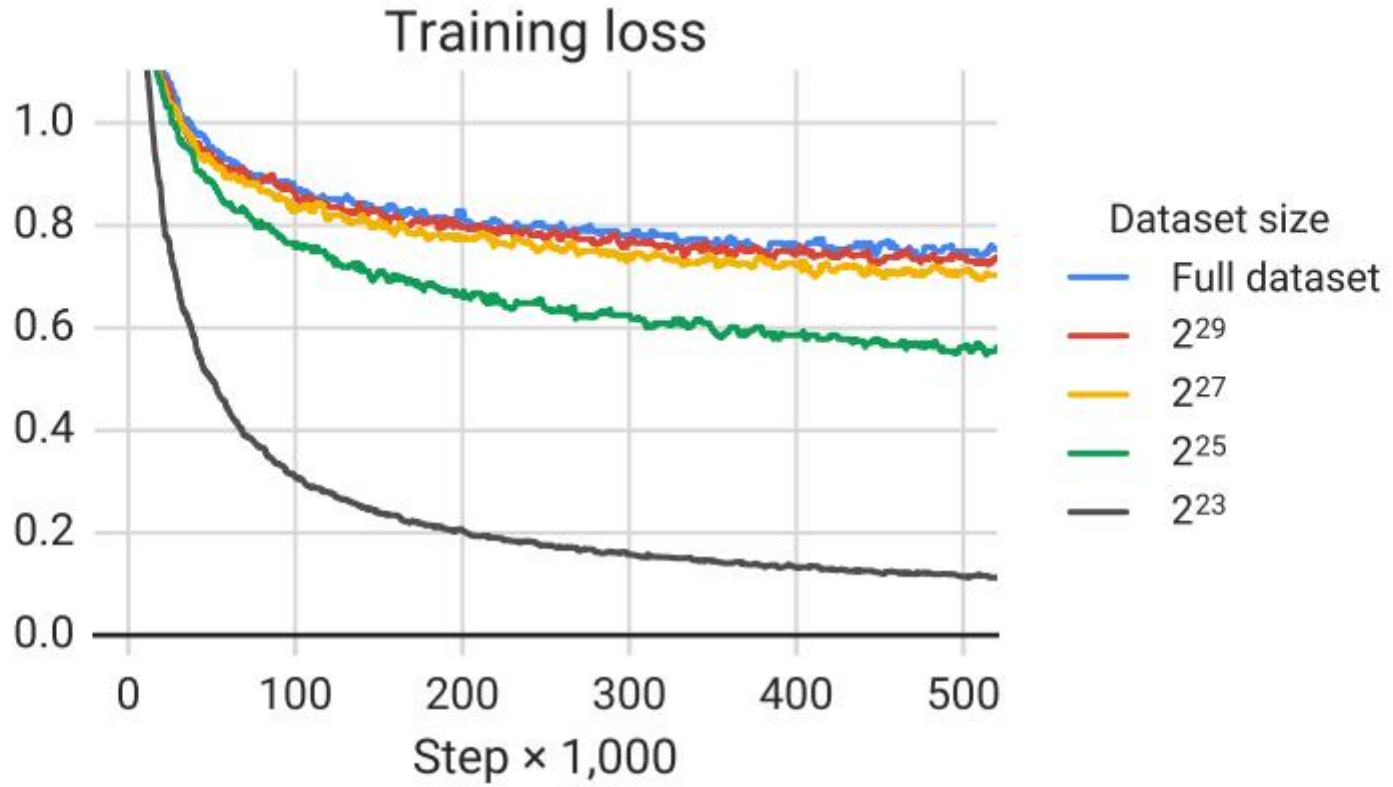


Figure 6: Pre-training loss for our original C4 data set as well as 4 artificially truncated versions. The sizes listed refer to the number of tokens in each data set. The four sizes considered correspond to repeating the data set between 64 and 4,096 times over the course of pre-training. Using a smaller data set size results in smaller training loss values, which may suggest some memorization of the unlabeled data set.

3.5. Training Strategy

3.5.1. Fine-Tuning Methods

- 모델의 모든 파라미터를 Fine-Tuning하는 것은 바람직하지 않음
- Adapter layers (Houlsby et al., 2019 등, 2025. 5. 15. 공부 예정): 기존 Transformer block 내 FFN 뒤에 dense-ReLU-dense 블록 삽입하여 사전학습된 파라미터 그대로 두고 adapter layer와 layer normalization 파라미터만 업데이트, SQuAD 등 저자원 과제에 작은 d 효과적
- Gradual unfreezing: 처음에는 일부 레이어만 조정하고 점진적으로 더 많은 레이어를 포함

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

Table 10: 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.

3.5. Training Strategy

3.5.2. Multi-Task Learning

- “Our goal is to not under- or over-train the model—that is, we want the model to see enough data from a given task that it can perform the task well, but not to see so much data that it memorizes the training set.” p. 31
- **Examples-proportional Mixing:** 과제 데이터셋 크기 비례, $r_m = \min(e_m, K)/\sum \min(e_n, K)$, K는 인위적 데이터셋 사이즈 상한선
- **Temperature-scaled Mixing:** 과제별 데이터셋 크기 차이 완화, $r'_m = r_m^{1/T}/\sum r_n^{1/T}$, T 클수록 비율 평준화(평탄화)
- **Equal Mixing:** 각 과제 균등 확률 샘플링

3.5.3. Combining Multi-Task Learning with Fine-Tuning

- **MT-DNN:** Pre-train on all tasks at once, fine-tune on individual supervised tasks
  - Pre-train on examples-proportional mixture with K = 2^{19}, fine-tune on each individual downstream task
  - Leave-One-Out
  - Pre-train on examples-proportional mixture of all supervised tasks

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

Table 11: 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 $K = 2^{21}$.

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**

Table 12: Comparison of unsupervised pre-training, multi-task learning, and various forms of multi-task pre-training.

번역 과제는 영어 기반 사전학습 효과가 상대적으로 적은 반면
나머지 과제들은 비지도 사전학습이 매우 중요한 역할

3.6. Scaling

- “Bitter lesson”(Sutton, 2019 등): “[G]eneral methods that can leverage additional computation ultimately win out against methods that rely on human expertise. … [I]t has repeatedly been shown that scaling up produces improved performance compared to more carefully-engineered methods.”
- “You were just given 4× more compute. How should you use it?”
  - 4× as many steps
  - 2× as many steps with the 2× bigger model
  - 4× bigger model
  - 앙상블 사용
- 결과
  - 학습시간이나 배치 사이즈를 4배씩 늘리는 것보다는 모델 크기 4배 늘리는 것이 우월
  - ‘학습시간 2배 × 모델 2배’와 ‘모델 4배’는 큰 차이 없었음

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

Table 13: 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.

3.7. Putting It All Together (T5)

- Objective: Denoising objective inspired by SpanBERT (Joshi et al., 2019)
- Longer Training: 1M steps, batch size 2^{11} sequences of length 512, 1T pre-training tokens (32× baseline)
- Model Sizes: Base 220M, Small 60M, Large 770M, 3B, 11B
- Multi-Task Pre-Training
- Fine-tuning on individual GLUE and SuperGLUE tasks
- Beam Search (Sutskever, Vinyals & Le, 2014. 9.)
- Results on Test Set (except SQuAD: Validation Set)

⟶ 24개 과제 중 18개에서 (논문 발표 당시) SOTA 달성

Model	GLUE Average	CoLA Matthew's	SST-2 Accuracy	MRPC F1	MRPC Accuracy	STS-B Pearson	STS-B Spearman
Previous best	89.4[a]	69.2[b]	97.1[a]	**93.6**[b]	**91.5**[b]	92.7[b]	92.3[b]
T5-Small	77.4	41.0	91.8	89.7	86.6	85.6	85.0
T5-Base	82.7	51.1	95.2	90.7	87.5	89.4	88.6
T5-Large	86.4	61.2	96.3	92.4	89.9	89.9	89.2
T5-3B	88.5	67.1	97.4	92.5	90.0	90.6	89.8
T5-11B	**90.3**	**71.6**	**97.5**	92.8	90.4	**93.1**	**92.8**

Model	QQP F1	QQP Accuracy	MNLI-m Accuracy	MNLI-mm Accuracy	QNLI Accuracy	RTE Accuracy	WNLI Accuracy
Previous best	74.8[c]	**90.7**[b]	91.3[a]	91.0[a]	**99.2**[a]	89.2[a]	91.8[a]
T5-Small	70.0	88.0	82.4	82.3	90.3	69.9	69.2
T5-Base	72.6	89.4	87.1	86.2	93.7	80.1	78.8
T5-Large	73.9	89.9	89.9	89.6	94.8	87.2	85.6
T5-3B	74.4	89.7	91.4	91.2	96.3	91.1	89.7
T5-11B	**75.1**	90.6	**92.2**	**91.9**	96.9	**92.8**	**94.5**

Model	SQuAD EM	SQuAD F1	SuperGLUE Average	BoolQ Accuracy	CB F1	CB Accuracy	COPA Accuracy
Previous best	90.1[a]	95.5[a]	84.6[d]	87.1[d]	90.5[d]	95.2[d]	90.6[d]
T5-Small	79.10	87.24	63.3	76.4	56.9	81.6	46.0
T5-Base	85.44	92.08	76.2	81.4	86.2	94.0	71.2
T5-Large	86.66	93.79	82.3	85.4	91.6	94.8	83.4
T5-3B	88.53	94.95	86.4	89.9	90.3	94.4	92.0
T5-11B	**91.26**	**96.22**	**88.9**	**91.2**	**93.9**	**96.8**	**94.8**

Model	MultiRC F1a	MultiRC EM	ReCoRD F1	ReCoRD Accuracy	RTE Accuracy	WiC Accuracy	WSC Accuracy
Previous best	84.4[d]	52.5[d]	90.6[d]	90.0[d]	88.2[d]	69.9[d]	89.0[d]
T5-Small	69.3	26.3	56.3	55.4	73.3	66.9	70.5
T5-Base	79.7	43.1	75.0	74.2	81.5	68.3	80.8
T5-Large	83.3	50.7	86.8	85.9	87.8	69.3	86.3
T5-3B	86.8	58.3	91.2	90.4	90.7	72.1	90.4
T5-11B	**88.1**	**63.3**	**94.1**	**93.4**	**92.5**	**76.9**	**93.8**

Model	WMT EnDe BLEU	WMT EnFr BLEU	WMT EnRo BLEU	CNN/DM ROUGE-1	CNN/DM ROUGE-2	CNN/DM ROUGE-L
Previous best	**33.8**[e]	**43.8**[e]	**38.5**[f]	43.47[g]	20.30[g]	40.63[g]
T5-Small	26.7	36.0	26.8	41.12	19.56	38.35
T5-Base	30.9	41.2	28.0	42.05	20.34	39.40
T5-Large	32.0	41.5	28.1	42.50	20.68	39.75
T5-3B	31.8	42.6	28.2	42.72	21.02	39.94
T5-11B	32.1	43.4	28.1	**43.52**	**21.55**	**40.69**

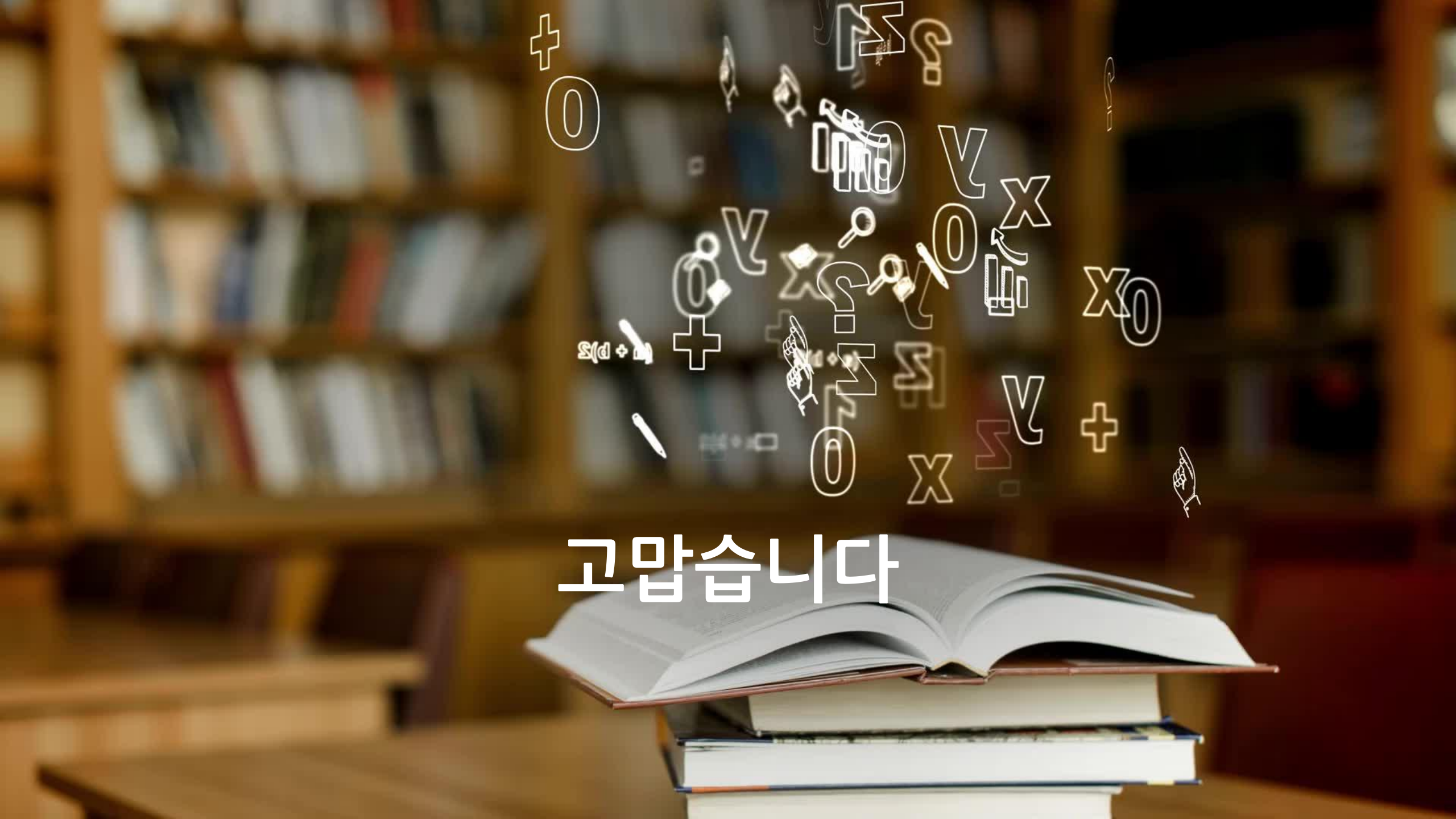
Table 14: Performance of our T5 variants on every task we study. Small, Base, Large, 3B, and 11B refer to model configurations with 60 million, 220 million, 770 million, 3 billion, and 11 billion parameters, respectively. In the first row of each table,

4.1. Takeaways

- **Text-to-Text:** 다양한 텍스트 작업을 동일한 형식으로 처리 가능
- **Architectures:** Encoder-Decoder 구조가 가장 우수, BERT 스타일보다 파라미터 많지만 계산 비용은 유사, Encoder-Decoder 파라미터 공유 시 성능 저하 거의 없음
- **Unsupervised Objectives:** 다양한 denoising objectives 비슷한 성능, 짧은 target sequence 생성하는 방식이 효율적
- **Data Sets:** C4 도입, 도메인 특화 데이터셋이 일부 태스크에서 유리하나 일반성 낮고 작음
- **Training Strategies:** 전체 파라미터 Fine-Tuning이 가장 좋은 성능, Multi-Task 학습은 비율 설정이 중요, Unsupervised + Supervised Fine-Tuning 조합이 효과적
- **Scaling:** 데이터 양, 모델 크기, 앙상블 모두 성능 향상에 기여, 큰 모델 짧게 학습한 결과가 종종 더 작고 오래 학습한 모델보다 우수
- **Pushing the Limits:** 11B 모델로 1T token 이상 학습, Multi-Task Pre-Training + 개별 과제 Fine-Tuning으로 (논문 발표 당시) SOTA 달성

4.2. Outlook

- **The Inconvenience of Large Models:** 모델 클수록 성능 좋지만 클라이언트 단 추론(Client-side Inference)이나 연합학습(Federated Learning)처럼 자원 제약 있는 환경에서 비효율적, 저비용 환경에서도 전이학습 활용할 수 있도록 Distillation, Parameter Sharing, Conditional Computation 같은 방법 연구 필요
- **More Efficient Knowledge Extraction:** 현재는 Span Denoising 방식이 주류지만 일반 지식 학습에 반드시 효과적인지 의문, 1T 토큰을 학습하지 않고도 우수한 성능 낼 수 있는 효율적 사전학습 방식 탐색 필요, 예: 진짜/가짜 텍스트 구별 학습(ELECTRA, Clark et. al., 2020)
- **Formalizing the Similarity between Tasks:** 사전학습 데이터와 다운스트림 과제 간 유사성이 성능에 영향, 유사성을 공식화하여 더 정교한 사전학습 데이터 선택 전략 마련 필요
- **Language-agnostic Models:** 영어 기반 사전학습 모델은 번역 과제에서 한계, 다양한 언어에 대응할 수 있는 모델 필요


고맙습니다