부스트캠프 AI Tech 2기

boostcamp ai tech

ELMo, GPT-1

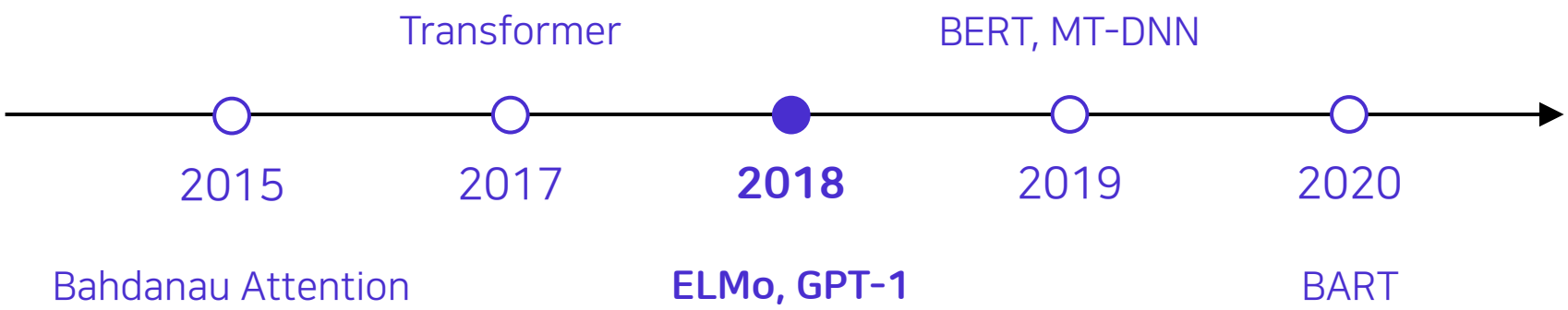
Deep contextualized word representations, NAACL 2018
Improving Language Understanding with Unsupervised Learning, Technical Report OpenAI, 2018

Email : jinmang2@gmail.com

GitHub : github.com/jinmang2

Huggingface Hub: huggingface.co/jinmang2

Boostcamp AI Tech 2 NLP 논문 모임에 오신 여러분 환영합니다!



00. Why ELMo and GPT-1?

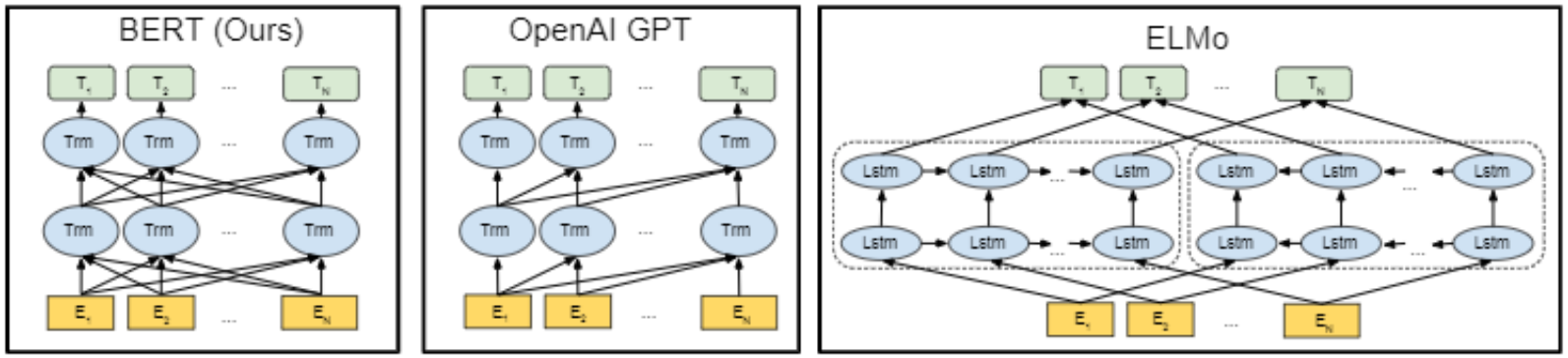


Figure 3: Differences in pre-training model architectures. BERT uses a bidirectional Transformer. OpenAI GPT uses a left-to-right Transformer. ELMo uses the concatenation of independently trained left-to-right and right-to-left LSTMs to generate features for downstream tasks. Among the three, only BERT representations are jointly conditioned on both left and right context in all layers. In addition to the architecture differences, BERT and OpenAI GPT are fine-tuning approaches, while ELMo is a feature-based approach.

BERT에서 Target으로 삼고 있는 Paper!

01. ELMo (Deep contextualized word representations)

Abstract

We introduce a new type of *deep contextualized* word representation that models both (1) complex characteristics of word use (e.g., syntax and semantics), and (2) how these uses vary across linguistic contexts (i.e., to model polysemy). Our word vectors are learned functions of the internal states of a deep bidirectional language model (biLM), which is pre-trained on a large text corpus. We show that these representations can be easily added to existing models and significantly improve the state of the art across six challenging NLP problems, including question answering, textual entailment and sentiment analysis. We also present an analysis showing that exposing the deep internals of the pre-trained network is crucial, allowing downstream models to mix different types of semi-supervision signals.

Deep Contextualized Word Representation 소개

- Deep! 깊은가보다
- Contextulaized Word 표현? Word2Vec, GloVe랑 관련이 있나?

특징!

- 단어의 복잡한 특징 모델링 (syntax, semantics)
- 다양한 언어적 맥락 상에서 어떻게 사용되는지 학습 (polysemy)

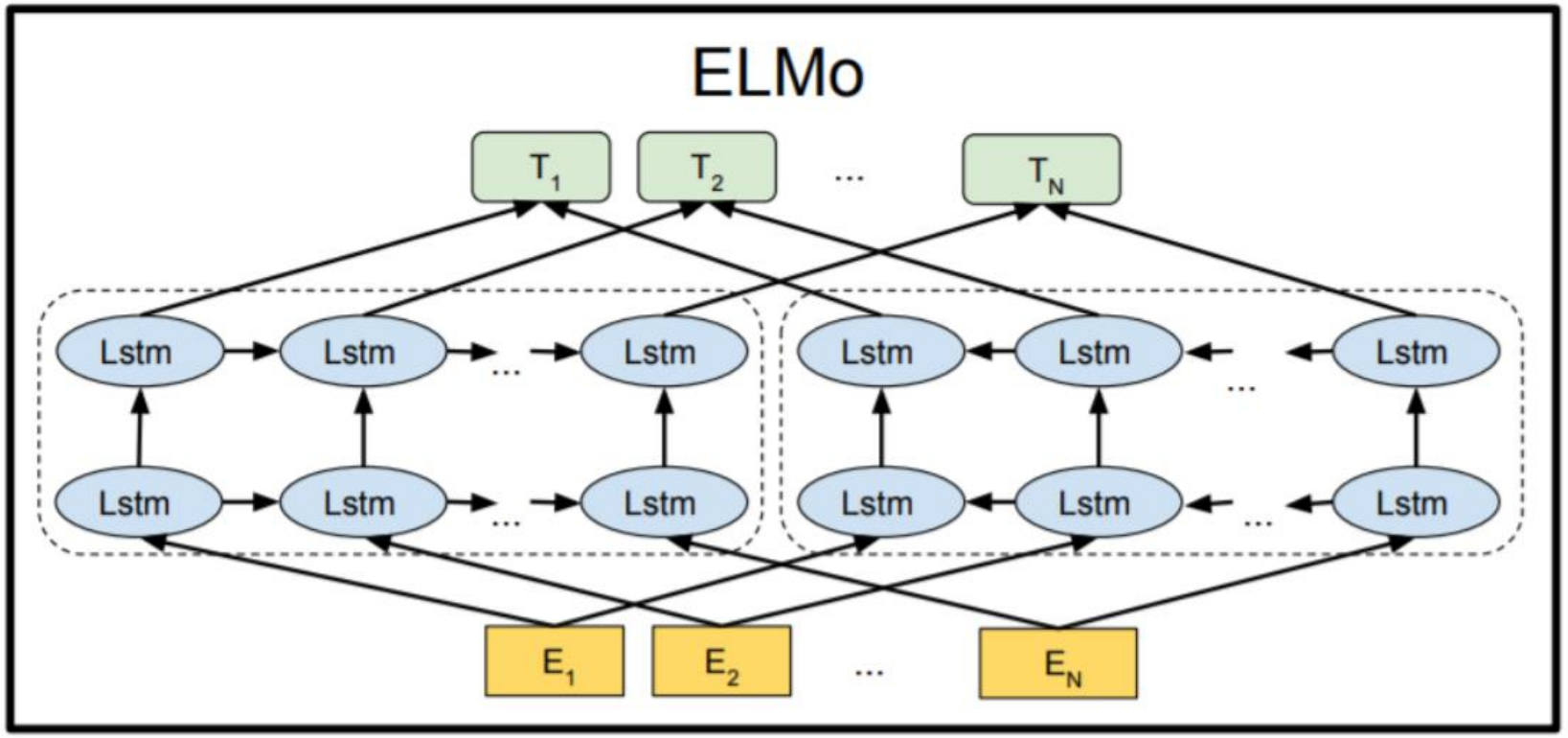
구조

- biLM @ large unlabeled corpora
- Char-CNN (for rich subword information)
- Feature-based pre-training (not fine-tuning)

당연하게도 SOTA! @ 6가지 Task

01. ELMo (Deep contextualized word representations)

High-Level View



01. ELMo (Deep contextualized word representations)

Related Work

Previous pre-trained vector is standard, but it is context-independent representations

- Large Scale Unlabeled Corpora로 학습한 Word2Vec, GloVe! QA, NLI, SRL에서 NLP 표준이 됨
- 그러나 이는 Context-Independentt한 표현만을 제공

Our apporoach has subword & multi-sense information

- 이전에 제안된 방법들로 위의 word vector의 단점을 어느 정도는 극복
  - Charagram: Embedding Words and Sentences via Character n-grams
  - Enriching Word Vectors with Subword Information
  - Efficient Non-parametric Estimation of Multiple Embeddings per Word in Vector Space
- 본 논문은 Char-CNN을 사용하여 subword unit으로부터 위와 동일한 이점을 얻을 수 있고
- Multi-sense information을 완벽하게 Downstream task에 전이

Deep contextual representation

- Context-Dependent한 표현을 학습하려던 연구도 많았음
- Context2Vec (Melamud et al., 2016)은 pivot word 주변의 맥락을 부호화하는데 biLSTM을 사용
- 아래의 paper도 비슷한 방식으로 context embedding을 학습하려고 시도했다.
  - CoVe (McCann et al., 2017)
  - Pre-ELMo (Peters et al., 2017)
- 그러나 위의 연구는 Parallel Corpora로 학습시킴. 더 많은 데이터로 학습시키면 당연히 좋아질 것
- 본 논문에선 대략 1B word benchmark (대략 30M sentences)의 monolingual 데이터로 biLM을 학습
- 그리고 본 연구는 위의 방법들을 일반화한 전략임 (Deep Contextual Representation)

Layer representations

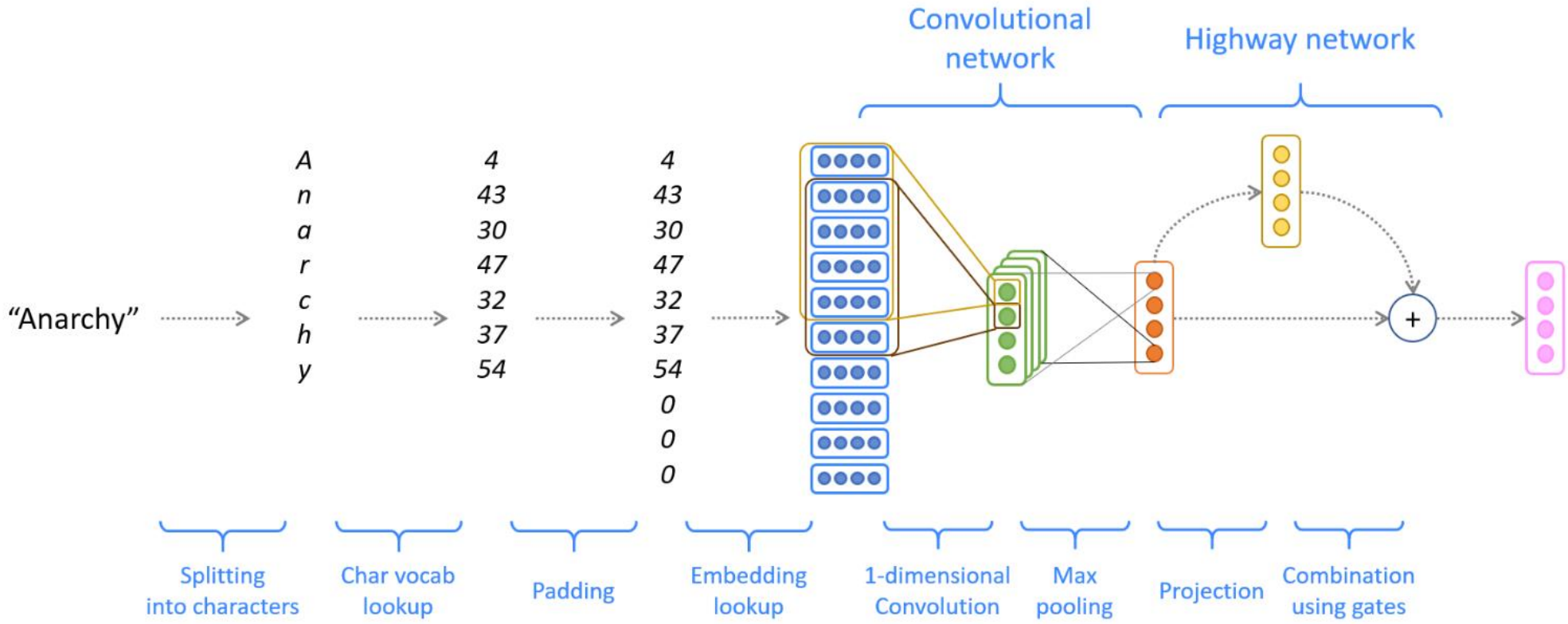
- Layer별 다른 표현을 학습한다고 보고한 여러 논문들이 있었다!
  - Dependency parsing (Hashimoto et el., 2017)
  - CCG super tagging (Søgaard and Goldberg, 2016)
- 본 논문에서 이에 대한 부분도 연구 실시! (linear combination of each layers representation)

Fix pre-train weights and Add additional task-specific model capacity

- Pre-train weights를 fix하고 task-specific model capacity를 추가!

01. ELMo (Deep contextualized word representations)

ELMo - CharCNN

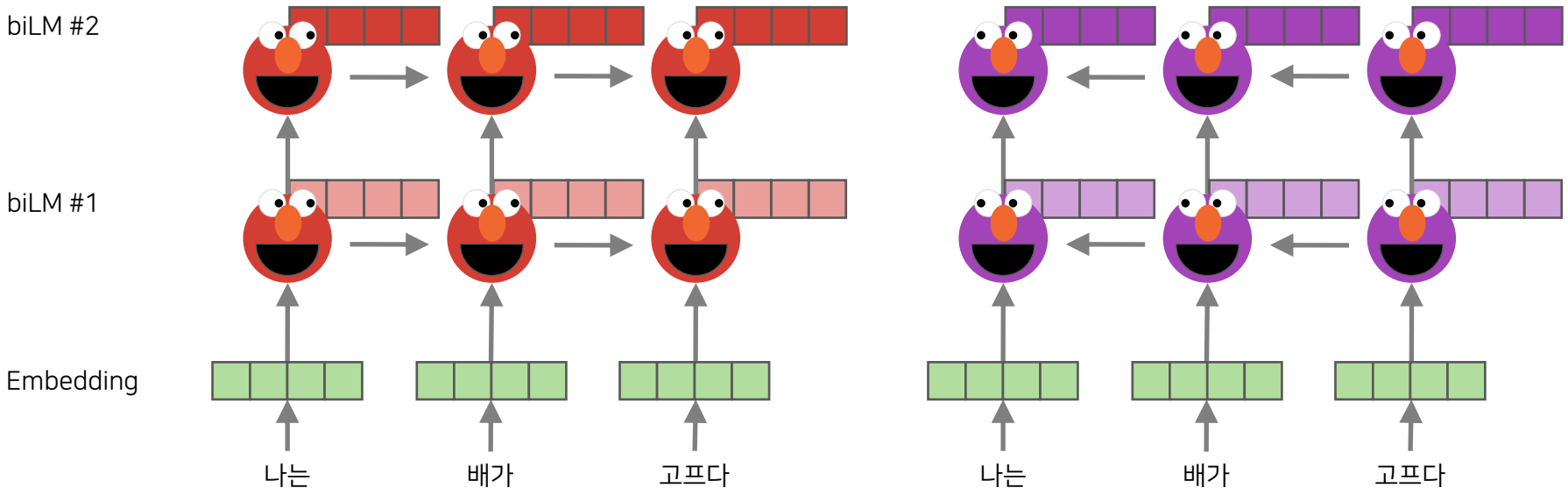


01. ELMo (Deep contextualized word representations)

Pre-training ELMo @ 1B Words

$$\sum_{k=1}^{N} \overbrace{(\log\ p(t_k|t_1,\ldots,t_{k-1};\Theta_x,\overrightarrow{\Theta}_{LSTM},\Theta_s)}^{forward\ LM} + \overbrace{\log\ p(t_k|t_{k+1},\ldots,t_N;\Theta_x,\overleftarrow{\Theta}_{LSTM},\Theta_s))}^{backward\ LM}$$

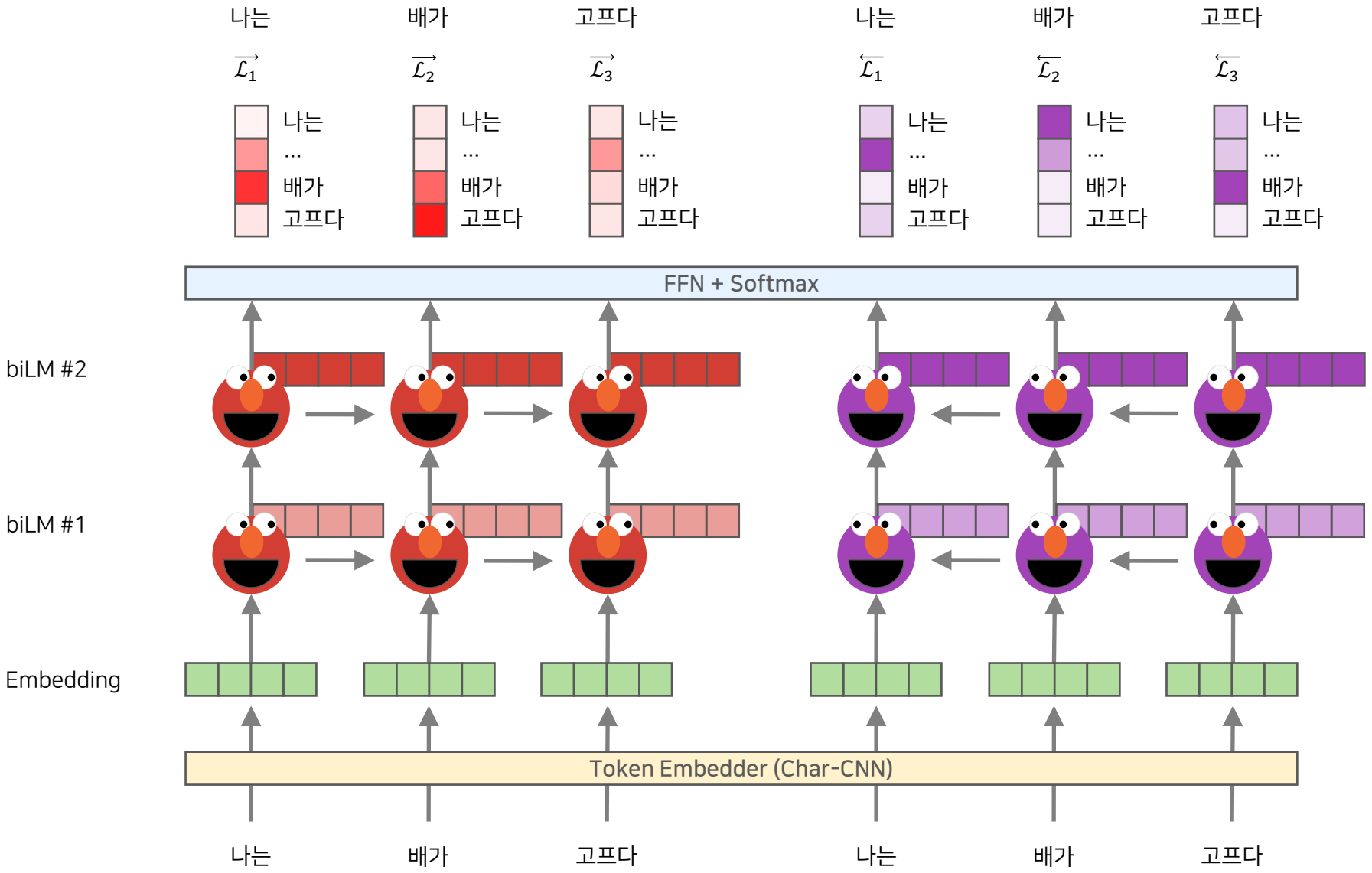
$$p(t_1,t_2,\ldots,t_N)=\prod_{k=1}^{N} p(t_k|t_1,t_2,\ldots,t_{k-1}) \qquad p(t_1,t_2,\ldots,t_N)=\prod_{k=1}^{N} p(t_k|t_{k+1},t_{k+2},\ldots,t_N)$$



https://jalammar.github.io/illustrated-bert/

01. ELMo (Deep contextualized word representations)

Pre-training ELMo @ 1B Words



https://jalammar.github.io/illustrated-bert/

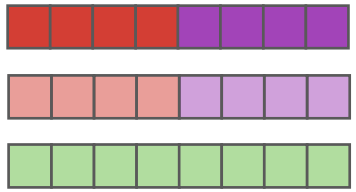
01. ELMo (Deep contextualized word representations)

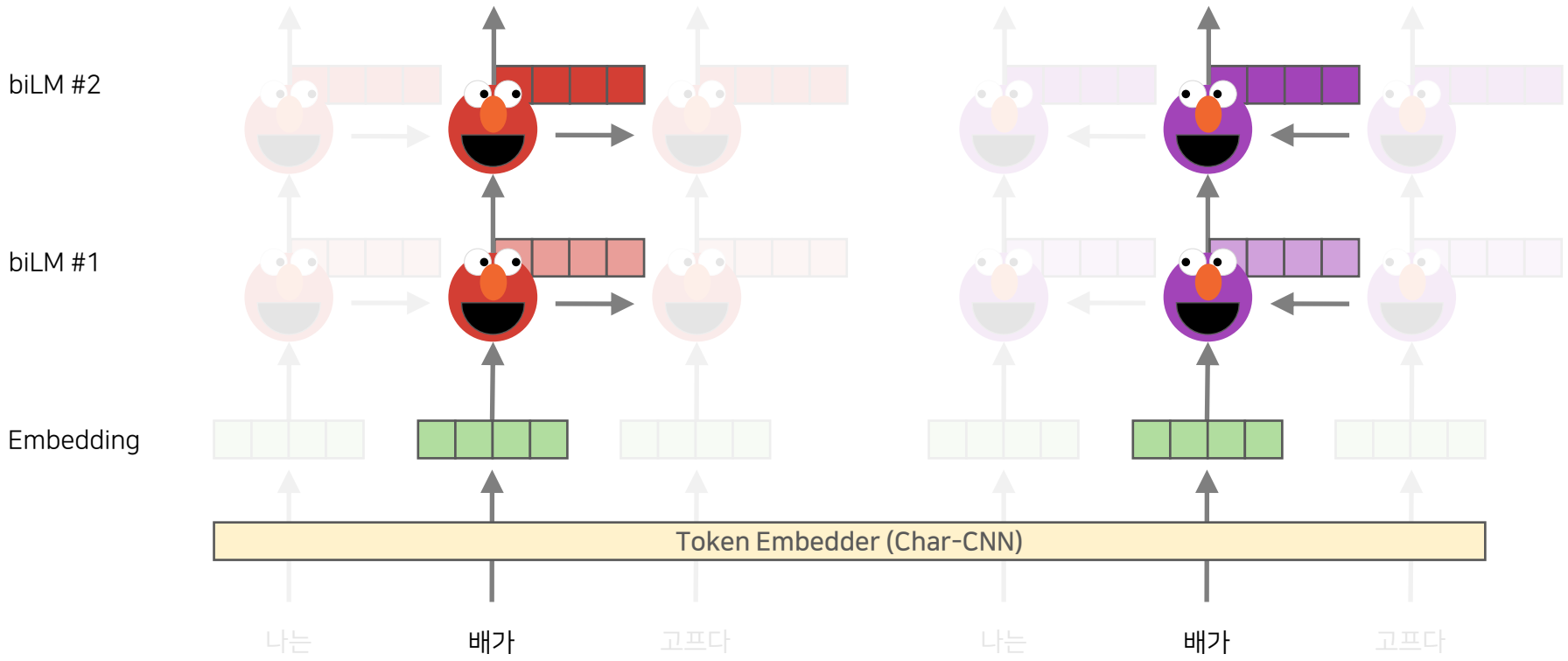
Fine-tune ELMo (feature-based)

Concatenate hidden layers

Multiply each vector by a weight on the task

Sum the (now weighted vectors)





https://jalammar.github.io/illustrated-bert/

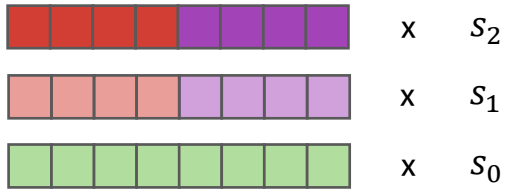
01. ELMo (Deep contextualized word representations)

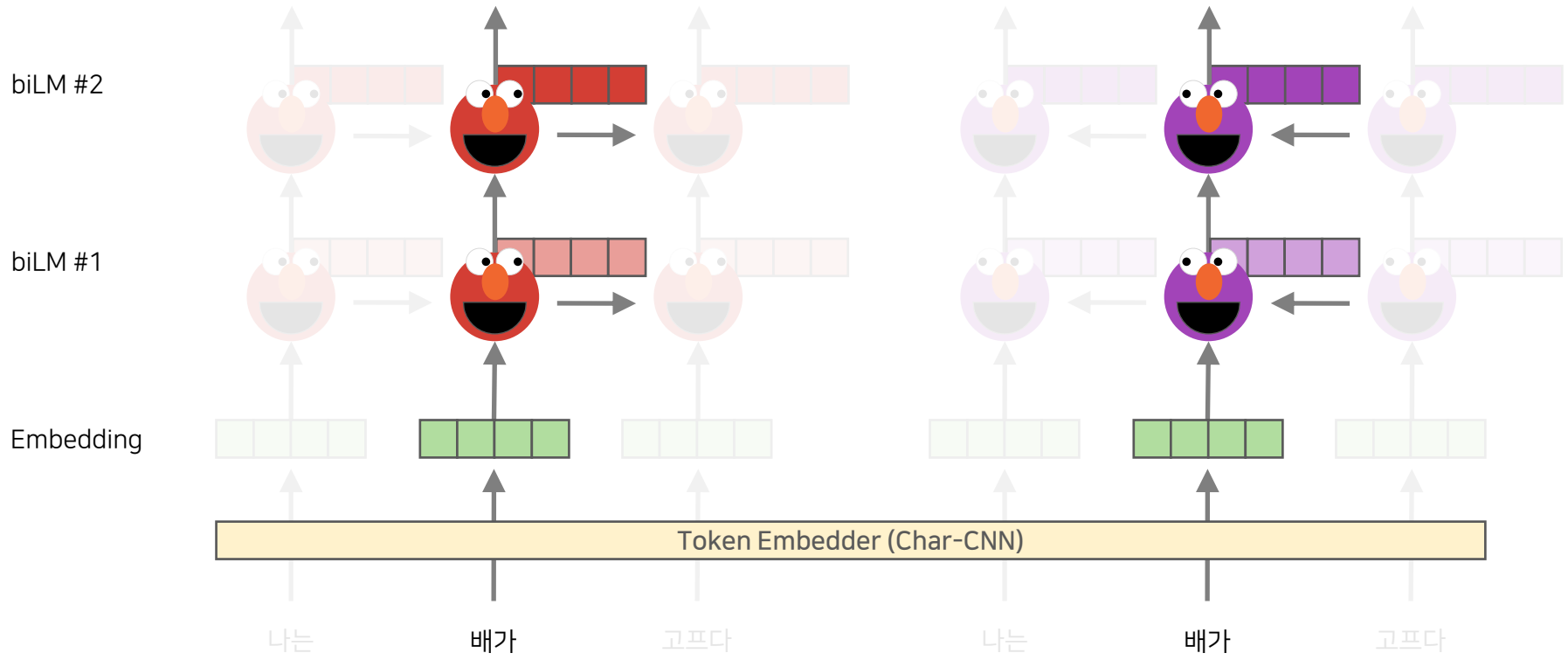
Fine-tune ELMo (feature-based)

Concatenate hidden layers

Multiply each vector by a weight on the task

Sum the (now weighted vectors)





https://jalammar.github.io/illustrated-bert/

01. ELMo (Deep contextualized word representations)

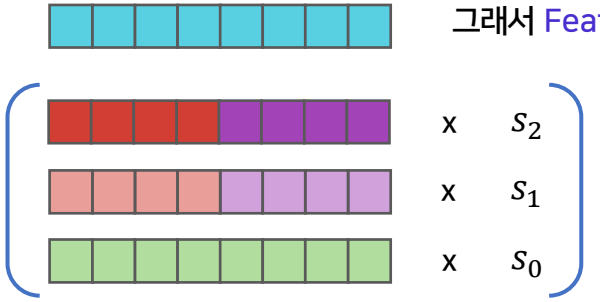
이게 ELMo Embedding이에요!
Fine-Tuning이랑은 조금 다르죠?
그래서 Feature-based라고 불러요!

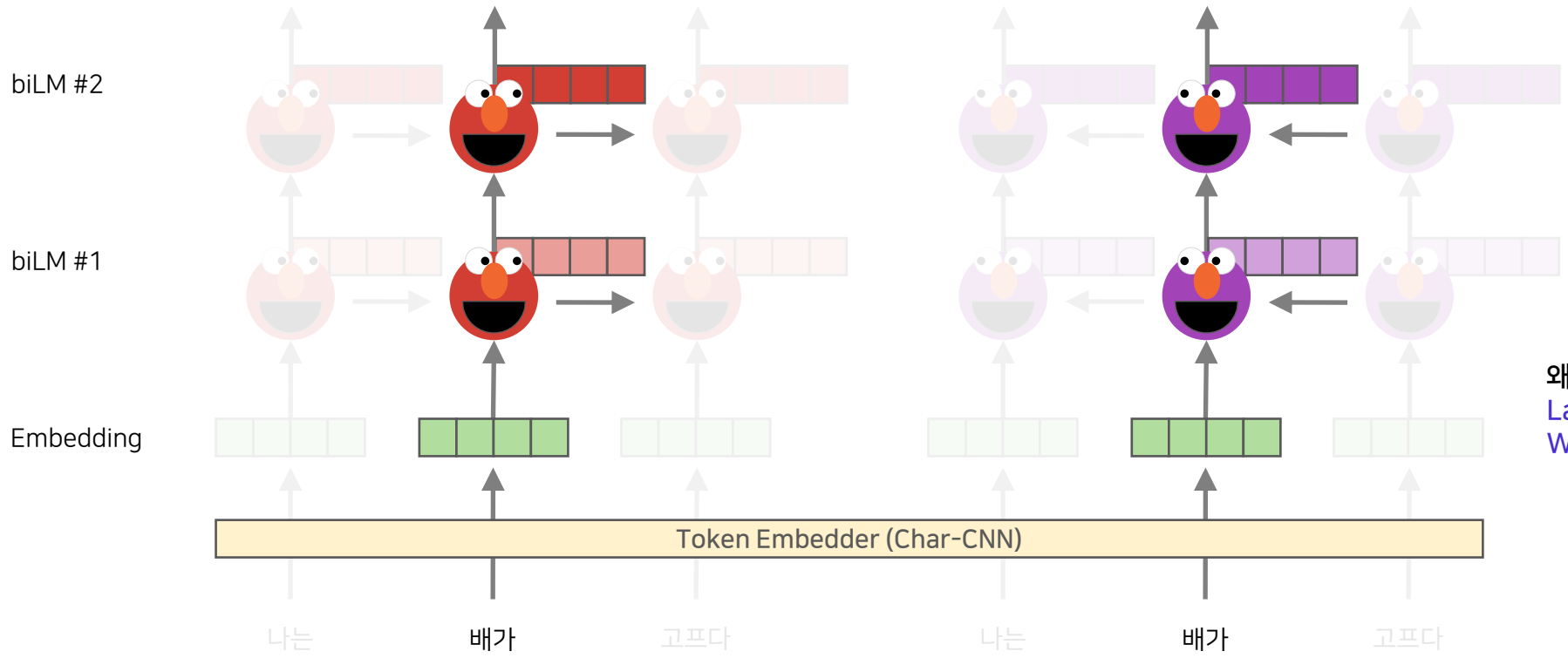
Fine-tune ELMo (feature-based)

Concatenate hidden layers

Multiply each vector by a weight on the task

Sum the (now weighted vectors)





왜 linear combination해줄까요?
Layer 별 학습 표현이 달라서 이걸
Weighted sum하면 더 좋다고 하네요!

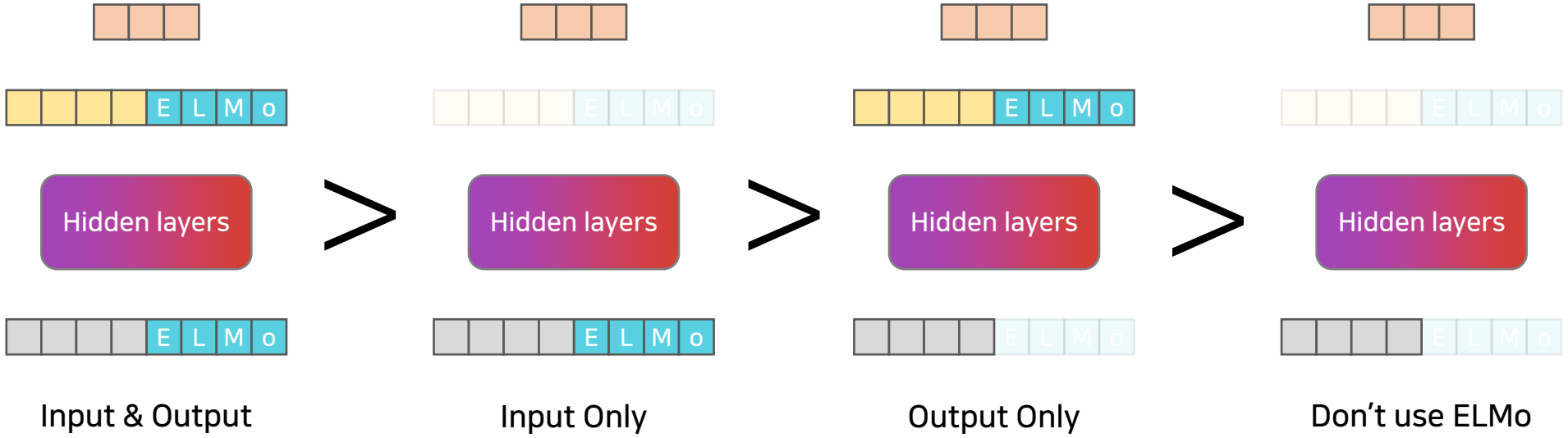
https://jalammar.github.io/illustrated-bert/

01. ELMo (Deep contextualized word representations)

Analysis: Where to include ELMo?

Task	Input Only	Input & Output	Output Only
SQuAD	85.1	**85.6**	84.8
SNLI	88.9	**89.5**	88.7
SRL	**84.7**	84.3	80.9

Table 3: Development set performance for SQuAD, SNLI and SRL when including ELMo at different locations in the supervised model.



https://www.youtube.com/watch?v=zV8klUwH32M

01. ELMo (Deep contextualized word representations)

ELMo's polesemy and results

TASK	PREVIOUS SOTA		OUR BASELINE	ELMO + BASELINE	INCREASE (ABSOLUTE/ RELATIVE)
SQuAD	Liu et al. (2017)	84.4	81.1	85.8	4.7 / 24.9%
SNLI	Chen et al. (2017)	88.6	88.0	88.7 ± 0.17	0.7 / 5.8%
SRL	He et al. (2017)	81.7	81.4	84.6	3.2 / 17.2%
Coref	Lee et al. (2017)	67.2	67.2	70.4	3.2 / 9.8%
NER	Peters et al. (2017)	91.93 ± 0.19	90.15	92.22 ± 0.10	2.06 / 21%
SST-5	McCann et al. (2017)	53.7	51.4	54.7 ± 0.5	3.3 / 6.8%

Table 1: Test set comparison of ELMo enhanced neural models with state-of-the-art single model baselines across six benchmark NLP tasks. The performance metric varies across tasks – accuracy for SNLI and SST-5; F_1 for SQuAD, SRL and NER; average F_1 for Coref. Due to the small test sizes for NER and SST-5, we report the mean and standard deviation across five runs with different random seeds. The "increase" column lists both the absolute and relative improvements over our baseline.

Task	Baseline	Last Only	All layers	
			λ=1	λ=0.001
SQuAD	80.8	84.7	85.0	**85.2**
SNLI	88.1	89.1	89.3	**89.5**
SRL	81.6	84.1	84.6	**84.8**

Table 2: Development set performance for SQuAD, SNLI and SRL comparing using all layers of the biLM (with different choices of regularization strength λ) to just the top layer.

01. ELMo (Deep contextualized word representations)

ELMo's polesemy and results

	Source	Nearest Neighbors
GloVe	play	playing, game, games, played, players, plays, player, Play, football, multiplayer
biLM	Chico Ruiz made a spectacular play on Alusik 's grounder {...}	Kieffer , the only junior in the group , was commended for his ability to hit in the clutch , as well as his all-round excellent play .
	Olivia De Havilland signed to do a Broadway play for Garson {...}	{...} they were actors who had been handed fat roles in a successful play , and had talent enough to fill the roles competently , with nice understatement .

Table 4: Nearest neighbors to "play" using GloVe and the context embeddings from a biLM.

Model	F_1
WordNet 1st Sense Baseline	65.9
Raganato et al. (2017a)	69.9
Iacobacci et al. (2016)	**70.1**
CoVe, First Layer	59.4
CoVe, Second Layer	*64.7*
biLM, First layer	67.4
biLM, Second layer	*69.0*

Table 5: All-words fine grained WSD F_1. For CoVe and the biLM, we report scores for both the first and second layer biLSTMs.

Model	Acc.
Collobert et al. (2011)	97.3
Ma and Hovy (2016)	97.6
Ling et al. (2015)	**97.8**
CoVe, First Layer	*93.3*
CoVe, Second Layer	92.8
biLM, First Layer	*97.3*
biLM, Second Layer	96.8

Table 6: Test set POS tagging accuracies for PTB. For CoVe and the biLM, we report scores for both the first and second layer biLSTMs.

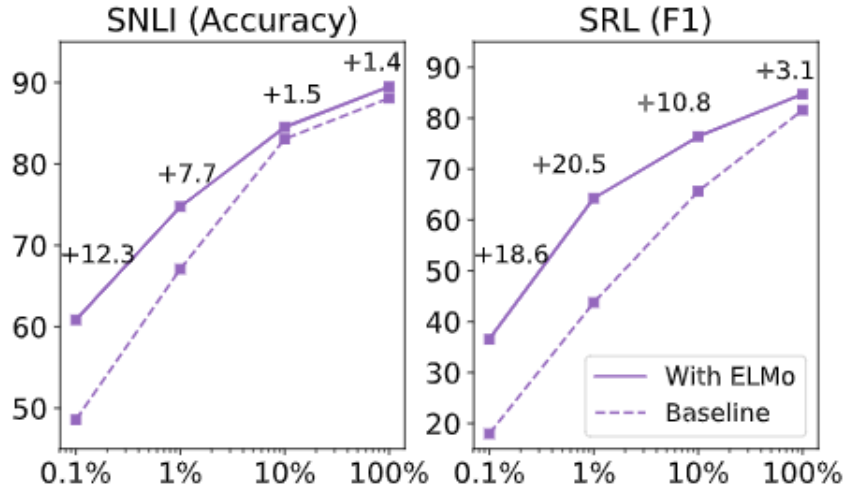


Figure 1: Comparison of baseline vs. ELMo performance for SNLI and SRL as the training set size is varied from 0.1% to 100%.

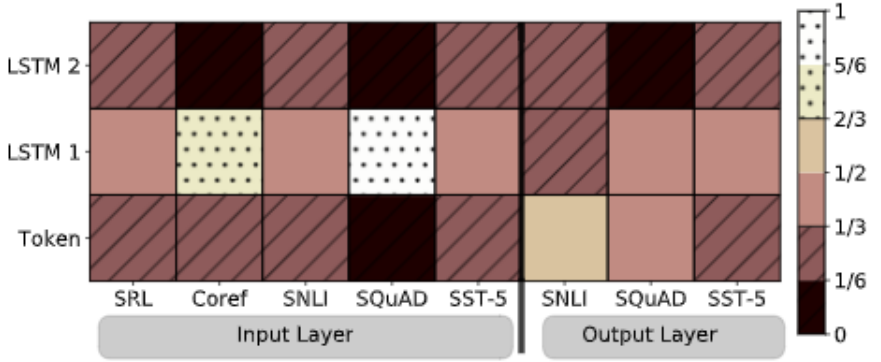
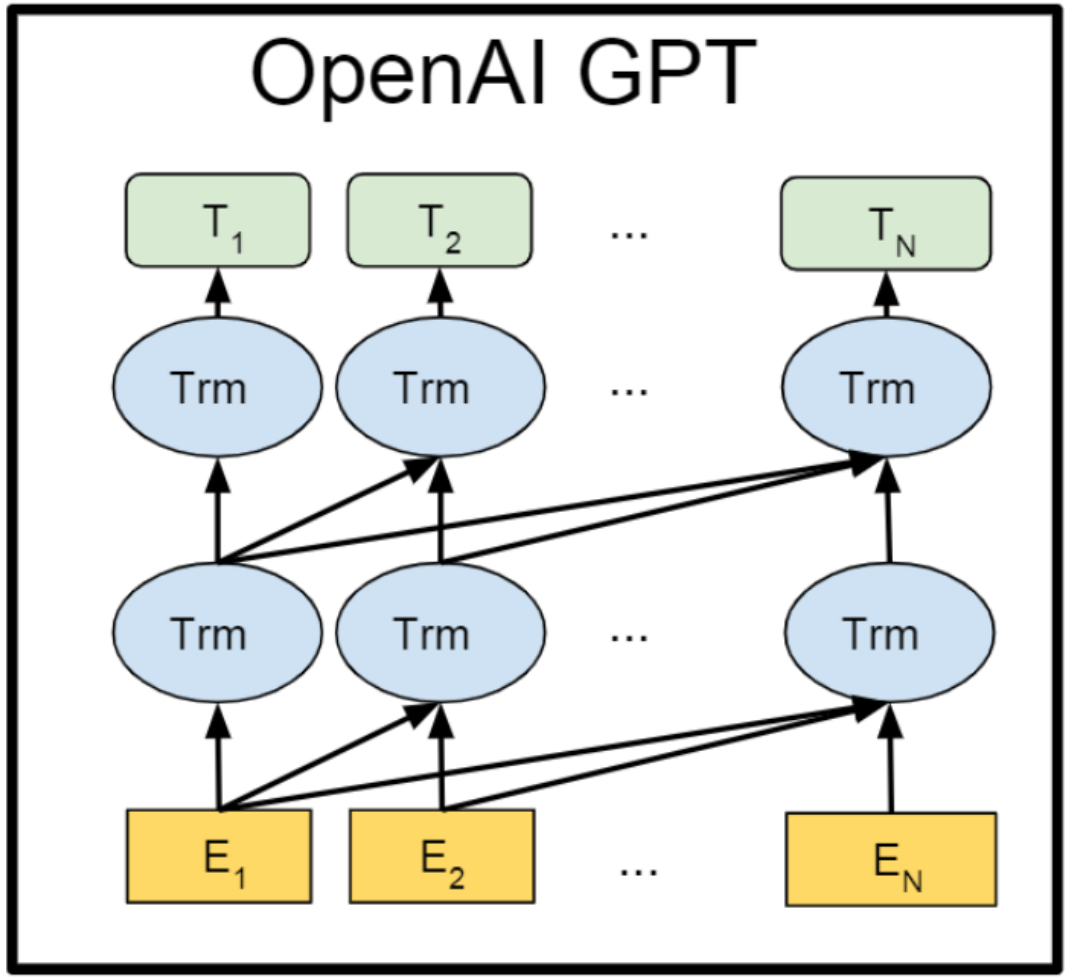


Figure 2: Visualization of softmax normalized biLM layer weights across tasks and ELMo locations. Normalized weights less then 1/3 are hatched with horizontal lines and those greater then 2/3 are speckled.

02. GPT-1 (Generative Pre-Training)

High-Level View



02. GPT-1 (Generative Pre-Training)

Abstract

Improving Language Understanding by Generative Pre-Training

Alec Radford OpenAI alec@openai.com

Karthik Narasimhan OpenAI karthikn@openai.com

Tim Salimans OpenAI tim@openai.com

Ilya Sutskever OpenAI ilyasu@openai.com

Abstract

Natural language understanding comprises a wide range of diverse tasks such as textual entailment, question answering, semantic similarity assessment, and document classification. Although large unlabeled text corpora are abundant, labeled data for learning these specific tasks is scarce, making it challenging for discriminatively trained models to perform adequately. We demonstrate that large gains on these tasks can be realized by *generative pre-training* of a language model on a diverse corpus of unlabeled text, followed by *discriminative fine-tuning* on each specific task. In contrast to previous approaches, we make use of task-aware input transformations during fine-tuning to achieve effective transfer while requiring minimal changes to the model architecture. We demonstrate the effectiveness of our approach on a wide range of benchmarks for natural language understanding. Our general task-agnostic model outperforms discriminatively trained models that use architectures specifically crafted for each task, significantly improving upon the state of the art in 9 out of the 12 tasks studied. For instance, we achieve absolute improvements of 8.9% on commonsense reasoning (Stories Cloze Test), 5.7% on question answering (RACE), and 1.5% on textual entailment (MultiNLI).

DATASET	TASK	SOTA	OURS
SNLI	Textual Entailment	89.3	**89.9**
MNLI Matched	Textual Entailment	80.6	**82.1**
MNLI Mismatched	Textual Entailment	80.1	**81.4**
SciTail	Textual Entailment	83.3	**88.3**
QNLI	Textual Entailment	82.3	**88.1**
RTE	Textual Entailment	**61.7**	56.0
STS-B	Semantic Similarity	81.0	**82.0**
QQP	Semantic Similarity	66.1	**70.3**
MRPC	Semantic Similarity	**86.0**	82.3
RACE	Reading Comprehension	53.3	**59.0**
ROCStories	Commonsense Reasoning	77.6	**86.5**
COPA	Commonsense Reasoning	71.2	**78.6**
SST-2	Sentiment Analysis	**93.2**	91.3
CoLA	Linguistic Acceptability	35.0	**45.4**
GLUE	Multi Task Benchmark	68.9	**72.8**

https://openai.com/blog/language-unsupervised/

02. GPT-1 (Generative Pre-Training)

Introduction

The ability to learn effectively from raw text is crucial to alleviating the dependence on supervised learning in natural language processing (NLP). Most deep learning methods require substantial amounts of manually labeled data, which restricts their applicability in many domains that suffer from a dearth of annotated resources [61]. In these situations, models that can leverage linguistic information from unlabeled data provide a valuable alternative to gathering more annotation, which can be time-consuming and expensive. Further, even in cases where considerable supervision is available, learning good representations in an unsupervised fashion can provide a significant performance boost. The most compelling evidence for this so far has been the extensive use of pre-trained word embeddings [10, 39, 42] to improve performance on a range of NLP tasks [8, 11, 26, 45].

Leveraging more than word-level information from unlabeled text, however, is challenging for two main reasons. First, it is unclear what type of optimization objectives are most effective at learning text representations that are useful for transfer. Recent research has looked at various objectives such as language modeling [44], machine translation [38], and discourse coherence [22], with each method outperforming the others on different tasks.[1] Second, there is no consensus on the most effective way to transfer these learned representations to the target task. Existing techniques involve a combination of making task-specific changes to the model architecture [43, 44], using intricate learning schemes [21] and adding auxiliary learning objectives [50]. These uncertainties have made it difficult to develop effective semi-supervised learning approaches for language processing.

- 일반적으로 Labeled Data는 부족함
- 하지만 Unlabeled Data는 많다!
  - 이를 활용할 수만 있다면 performance boost가 가능
  - Word2Vec, GloVe에서 이미 그 사례를 봤음
- 하지만 Unlabeled text에서 word-level information을 다루는 데에는 두 가지 문제점이 있다.
  1. 어떻게 학습해야 할까요?
     - Pre-train objective에 대한 고민
  2. 어떻게 전이 시켜야 할까요?
     - ELMo 방식 (task-specific model architecture)
     - ULMFiT 방식 (intricate learning schemes)
     - Add auxiliary learning objectives

02. GPT-1 (Generative Pre-Training)

Introduction

In this paper, we explore a semi-supervised approach for language understanding tasks using a combination of unsupervised pre-training and supervised fine-tuning. Our goal is to learn a universal representation that transfers with little adaptation to a wide range of tasks. We assume access to a large corpus of unlabeled text and several datasets with manually annotated training examples (target tasks). Our setup does not require these target tasks to be in the same domain as the unlabeled corpus. We employ a two-stage training procedure. First, we use a language modeling objective on the unlabeled data to learn the initial parameters of a neural network model. Subsequently, we adapt these parameters to a target task using the corresponding supervised objective.

For our model architecture, we use the *Transformer* [62], which has been shown to perform strongly on various tasks such as machine translation [62], document generation [34], and syntactic parsing [29]. This model choice provides us with a more structured memory for handling long-term dependencies in text, compared to alternatives like recurrent networks, resulting in robust transfer performance across diverse tasks. During transfer, we utilize task-specific input adaptations derived from traversal-style approaches [52], which process structured text input as a single contiguous sequence of tokens. As we demonstrate in our experiments, these adaptations enable us to fine-tune effectively with minimal changes to the architecture of the pre-trained model.

We evaluate our approach on four types of language understanding tasks – natural language inference, question answering, semantic similarity, and text classification. Our general task-agnostic model outperforms discriminatively trained models that employ architectures specifically crafted for each task, significantly improving upon the state of the art in 9 out of the 12 tasks studied. For instance, we achieve absolute improvements of 8.9% on commonsense reasoning (Stories Cloze Test) [40], 5.7% on question answering (RACE) [30], 1.5% on textual entailment (MultiNLI) [66] and 5.5% on the recently introduced GLUE multi-task benchmark [64]. We also analyzed zero-shot behaviors of the pre-trained model on four different settings and demonstrate that it acquires useful linguistic knowledge for downstream tasks.

- 뭔가 중구난방이고 합의된 Consensus가 없던 상황
- 본 논문에서 Semi-Supervised Approach 정리!
  - Pre-training and Fine-Tuning
  - 처음 나온 개념은 아님. 일반화 느낌
- 이전에 사용하던 LSTM을 버리고
- Transformer의 Decoder를 사용하여 Self-Attn의 이점을 가짐
- Transfer에선 DeepMind의 Reasoning about entailment with neural attention이라는 논문에서 영감을 얻음
- 즉, Traversal style인 task-specific input adaptation 사용
  - Structured text input을 single contiguous of tokens로 처리
- 실험 결과 최소한의 parameter 추가로 9/12 SOTA 달성!

02. GPT-1 (Generative Pre-Training)

Framework – 2 Stage

stage 1. Unsupervised pre-training

$$\mathcal{L}_1 = \sum_i \log P(u_i | u_{i-k}, \ldots, u_{i-1}; \Theta)$$

위 수식을 Transformer Decoder로 모델링!

$$\begin{aligned} h_o &= UW_e + W_p \\ h_1 &= \text{transformer\_block}(h_{l-1}) \; \forall i \in [1, n] \\ P(u) &= \text{softmax}(h_n W_e^T) \end{aligned}$$

- U는 context vector of tokens, 즉 `nn.Embedding`을 통과한 상태! (수식만 보면)
- n은 number of layers
- W_e는 token embedding matrix인데 `nn.Linear`로 구현하면 될 듯
- W_p는 position embedding matrix
- Transformer Decoder Block: Cross-Attention 제거하고 사용

stage 2. Supervised fine-tuning

$$P(y | x^1, \ldots, x^m) = \text{softmax}(h_l^m W_y)$$

- $x^1, \ldots, x^m$: 길이가 m인 input tokens
- h_l^m: final transformer decoder output of last token m
- W_y: task-specific layer. 이 부분만 유일하게 추가된다! (ELMo와 이 부분이 다름)

아래 수식을 optimize

$$\mathcal{L}_2 = \sum_{x,y} \log P(y | x^1, \ldots, x^m)$$

Final loss: Sum over

$$\mathcal{L} = \mathcal{L}_1 + \lambda * \mathcal{L}_2$$

- LM Loss를 사용하면 일반화 + 빠른 수렴 가능! (in Pre-ELMo paper)

02. GPT-1 (Generative Pre-Training)

Task-specific input transformations

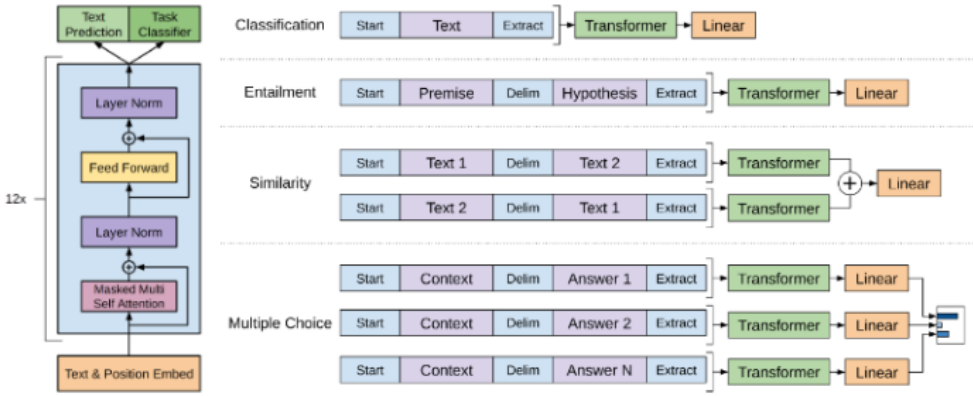


Figure 1: **(left)** Transformer architecture and training objectives used in this work. **(right)** Input transformations for fine-tuning on different tasks. We convert all structured inputs into token sequences to be processed by our pre-trained model, followed by a linear+softmax layer.

Table 1: A list of the different tasks and datasets used in our experiments.

Task	Datasets
Natural language inference	SNLI [5], MultiNLI [66], Question NLI [64], RTE [4], SciTail [25]
Question Answering	RACE [30], Story Cloze [40]
Sentence similarity	MSR Paraphrase Corpus [14], Quora Question Pairs [9], STS Benchmark [6]
Classification	Stanford Sentiment Treebank-2 [54], CoLA [65]

사실 상 GPT-1의 핵심! input을 어떻게 넣어줄 것이냐? Main figure보면 알 수 있음. 간단하게 정리

참고: 이는 DeepMind의 `Reasoning about entailment with neural attention`에서 영감을 얻었다고 함

- Text Classification: `<s> sentence <e>`
- Text entailment: `<s> premise $ hypothesis <e>`
- Similarity: `<s> sentence A $ sentence B <e>`, `<s> sentence B $ sentence C <e>`
- Question Answering and Commonsense Reasoning: `<s> z q & a_1 <e>`, ..., `<s> z q & a_k <e>`
  - `z`: document context, `q`: question, `a_k`: possible answer

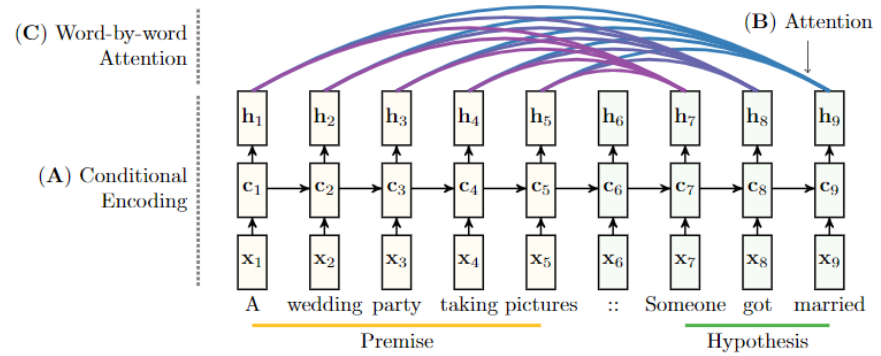


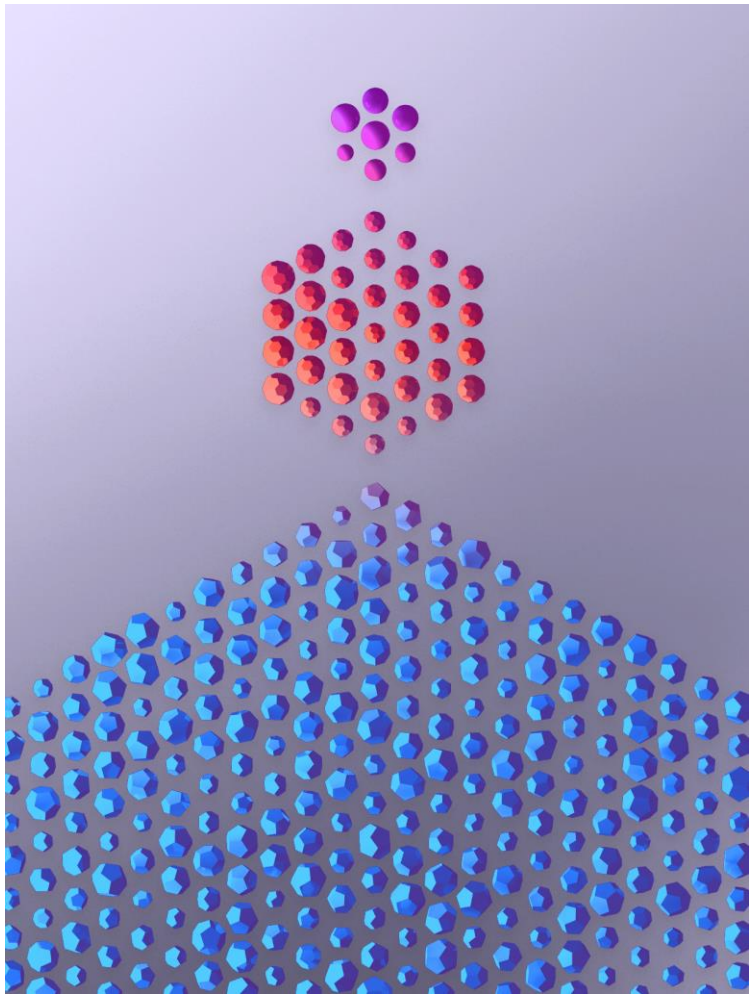
Figure 1: Recognizing textual entailment using (**A**) conditional encoding via two LSTMs, one over the premise and one over the hypothesis conditioned on the representation of the premise (c_5), (**B**) attention only based on the last output vector (h_9) or (**C**) word-by-word attention based on all output vectors of the hypothesis (h_7, h_8 and h_9).

https://arxiv.org/pdf/1509.06664.pdf

02. GPT-1 (Generative Pre-Training)

Experiment setup

- BookCorpus dataset 7,000 unique unpublished books (대략 800M words)
- Transformer Decoder (no cross-attention)
- 768 dimension, 12 heads, 3072 FFN dims
- Adam Optimizer with max learning rate 2.5e-04, cosine scheduler warm-up step 2000
- 100 epochs, mini-batch size 64, mas token length 512
- LayerNorm ~ N(0, 0.02)
- BPE vocab 40,000 merges
- Residual, embedding, and attention dropouts with rate of 0.1
- L2 Regularization with w = 0.01 (weight decay)
- Use GELU activation function
- Use learned position embedding
- Use ftfy library for clean text
- Use spaCy tokenizer for standardize some punctuation and whitespace



Fine tune setup

- Dropout 0.1, Learning rate 6.25e-5, Batch size 32
- 3 epochs, warm-up ratio 0.2% of training
- Linear learning rate scheduler

02. GPT-1 (Generative Pre-Training)

Results

Table 2: Experimental results on natural language inference tasks, comparing our model with current state-of-the-art methods. 5x indicates an ensemble of 5 models. All datasets use accuracy as the evaluation metric.

Method	MNLI-m	MNLI-mm	SNLI	SciTail	QNLI	RTE
ESIM + ELMo [44] (5x)	-	-	89.3	-	-	-
CAFE [58] (5x)	80.2	79.0	89.3	-	-	-
Stochastic Answer Network [35] (3x)	80.6	80.1	-	-	-	-
CAFE [58]	78.7	77.9	88.5	83.3		
GenSen [64]	71.4	71.3	-	-	82.3	59.2
Multi-task BiLSTM + Attn [64]	72.2	72.1	-	-	82.1	**61.7**
Finetuned Transformer LM (ours)	**82.1**	**81.4**	**89.9**	**88.3**	**88.1**	56.0

Table 3: Results on question answering and commonsense reasoning, comparing our model with current state-of-the-art methods.. 9x means an ensemble of 9 models.

Method	Story Cloze	RACE-m	RACE-h	RACE
val-LS-skip [55]	76.5	-	-	-
Hidden Coherence Model [7]	77.6	-	-	-
Dynamic Fusion Net [67] (9x)	-	55.6	49.4	51.2
BiAttention MRU [59] (9x)	-	60.2	50.3	53.3
Finetuned Transformer LM (ours)	**86.5**	**62.9**	**57.4**	**59.0**

02. GPT-1 (Generative Pre-Training)

Results

Table 4: Semantic similarity and classification results, comparing our model with current state-of-the-art methods. All task evaluations in this table were done using the GLUE benchmark. (*mc*= Mathews correlation, *acc*=Accuracy, *pc*=Pearson correlation)

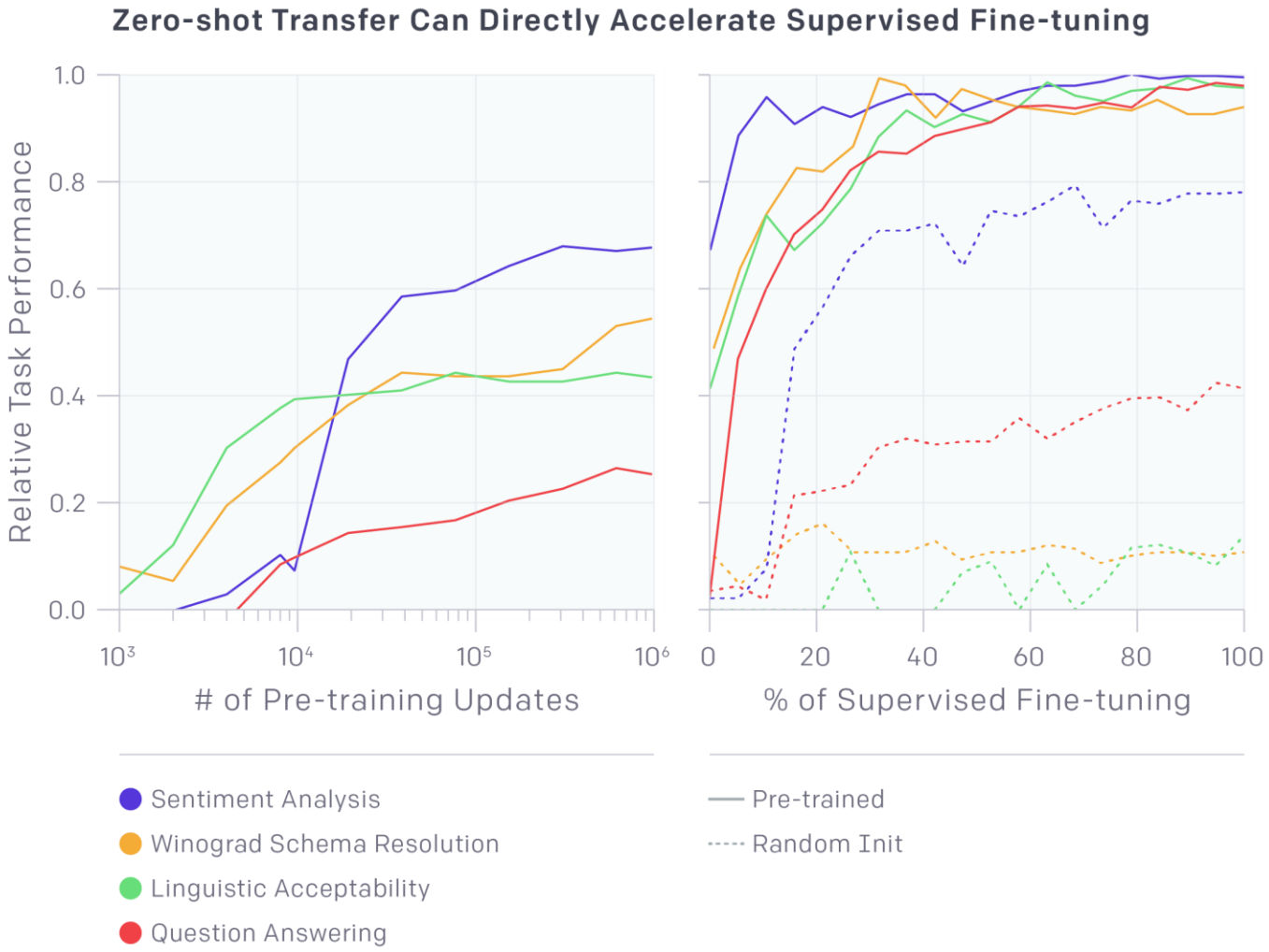
Method	Classification		Semantic Similarity			GLUE
	CoLA (mc)	SST2 (acc)	MRPC (F1)	STSB (pc)	QQP (F1)	
Sparse byte mLSTM [16]	-	**93.2**	-	-	-	-
TF-KLD [23]	-	-	**86.0**	-	-	-
ECNU (mixed ensemble) [60]	-	-	-	81.0	-	-
Single-task BiLSTM + ELMo + Attn [64]	35.0	90.2	80.2	55.5	66.1	64.8
Multi-task BiLSTM + ELMo + Attn [64]	18.9	91.6	83.5	72.8	63.3	68.9
Finetuned Transformer LM (ours)	**45.4**	91.3	82.3	**82.0**	**70.3**	**72.8**

Table 5: Analysis of various model ablations on different tasks. Avg. score is a unweighted average of all the results. (*mc*= Mathews correlation, *acc*=Accuracy, *pc*=Pearson correlation)

Method	Avg. Score	CoLA (mc)	SST2 (acc)	MRPC (F1)	STSB (pc)	QQP (F1)	MNLI (acc)	QNLI (acc)	RTE (acc)
Transformer w/ aux LM (full)	74.7	45.4	91.3	82.3	82.0	**70.3**	**81.8**	**88.1**	**56.0**
Transformer w/o pre-training	59.9	18.9	84.0	79.4	30.9	65.5	75.7	71.2	53.8
Transformer w/o aux LM	**75.0**	**47.9**	**92.0**	**84.9**	**83.2**	69.8	81.1	86.9	54.4
LSTM w/ aux LM	69.1	30.3	90.5	83.2	71.8	68.1	73.7	81.1	54.6

02. GPT-1 (Generative Pre-Training)

Results



부스트캠프 AI Tech 2기

boostcamp ai tech

Discussion

Email : jinmang2@gmail.com

GitHub : github.com/jinmang2

Huggingface Hub: huggingface.co/jinmang2

진명훈_T2216