BERT: Self-supervised learning meets Transformer

2022.01.11 @ SSL Study, MARG

Speaker: 박승원

BERT @ Google Search

https://en.wikipedia.org › wiki › BERT_(language_mod...

BERT (language model) - Wikipedia

Bidirectional Encoder Representations from Transformers (**BERT**) is a transformer-based machine learning technique for **natural language processing** (**NLP**) ...

Architecture · Analysis · History

https://arxiv.org › cs

BERT: Pre-training of Deep Bidirectional Transformers for ...

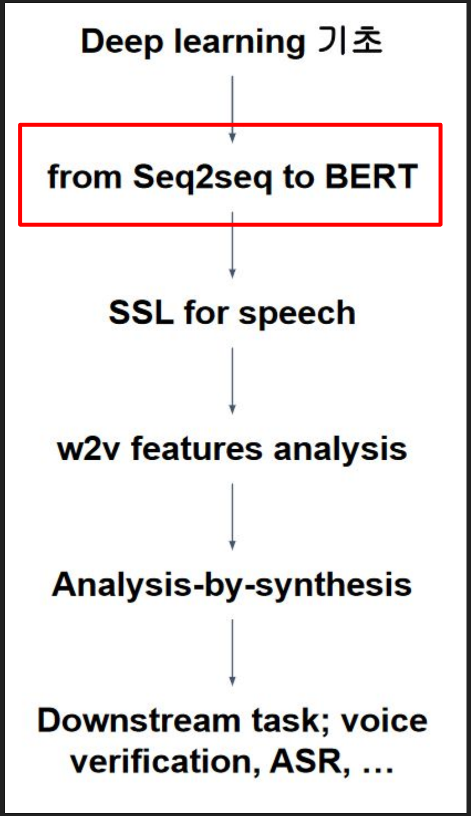
by J Devlin · 2018 · Cited by 31812 — Unlike recent language representation models, **BERT** is designed to pre-train deep bidirectional representations from unlabeled text by jointly ...

Why we'll be talking about BERT?

Last time: Basics of Deep Learning.
Next time: SSL for speech (wav2vec 2.0, CPC, …)

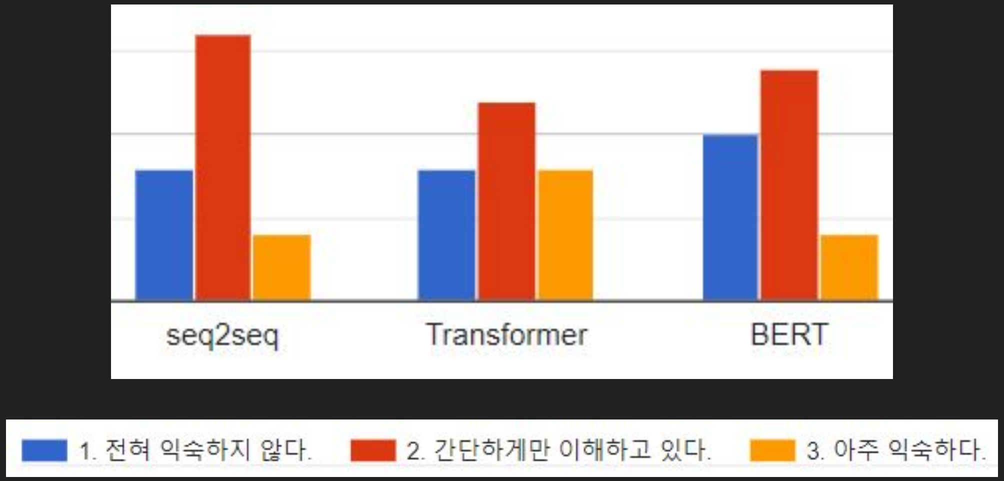
This time: Some prerequisites for the next time
(Transformer, Self-supervised learning)

Transformer + SSL = BERT
(we won't focus on NLP applications)
만들어놓고 보니 Transformer 슬라이드가 과반…



To make sure we're on the same page

We'll assume that the audience have no knowledge on:
(1) attention mechanisms, and (2) BERT



Contents

- Brief history of attention mechanism
  - seq2seq
  - vanilla attention with RNNs for NMT
  - transformer
- Self-supervised learning
  - why it matters?
  - examples of pretext tasks
- BERT: Self-supervised learning meets Transformer
  - impact, the good news & bad news
- Conclusion with recap questions
  - useful links, references

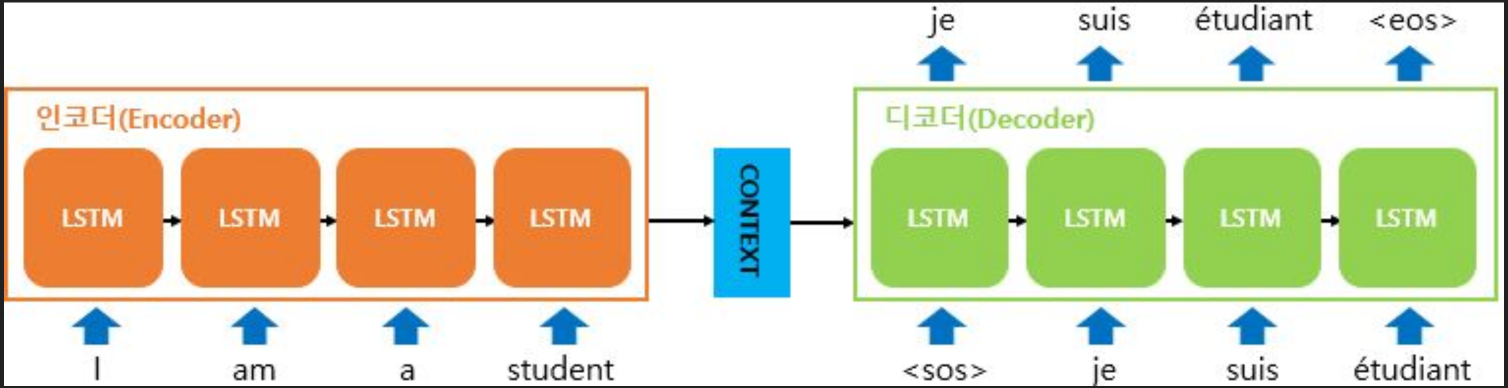
Brief history of attention mechanism

with some RNN basics

Some keywords

- RNN = Recurrent Neural Network
    - family of RNNs include: RNN (1986), LSTM (1997), GRU (2014)
- NMT = Neural Machine Translation (기계번역)
- Tokenization (in NLP)
    - “나는 사과를 먹었다” -> [‘나’, ‘#는’, ‘사과’, ‘#를’, ‘먹어’, ‘#ㅆ다’]
    - “Byte-Pair Encoding” is most commonly used (won’t address)
- Auto-regressive model
    - does not necessarily imply using RNNs; other architecture could be used. (e.g. WaveNet)
    - branch of generative models (perhaps the easiest one)
- Teacher-forcing
    - widely used training strategy; use ground-truth token for the next step.

sequence-to-sequence (seq2seq, 2014)



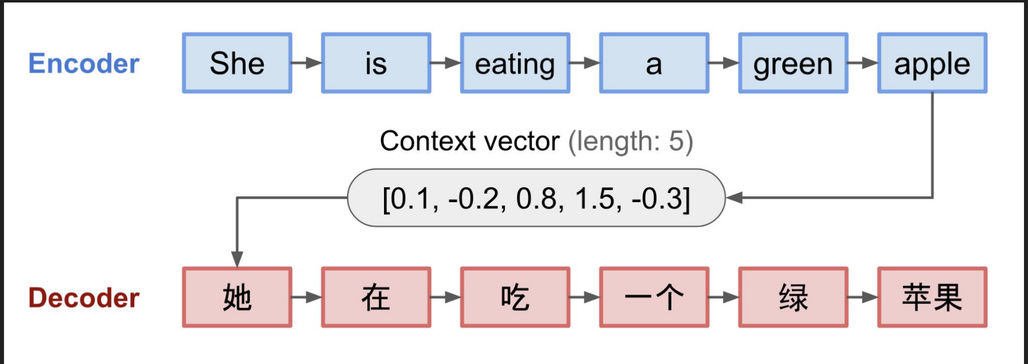
- Influenced by previous work, which is now called as ‘decoder’
  - Generating sequences with RNNs (Graves *et al.*, 2013)

Figure from https://wikidocs.net/24996

Problem of seq2seq

1. Need to compress length-varying data into a fixed-size “context vector”
2. Gradient vanishing due to long-term dependency

Those might cause the model to even fail with “copy-paste” task



3.3 Reversing the Source Sentences

While the LSTM is capable of solving problems with long term dependencies, we discovered that the LSTM learns much better when the source sentences are reversed (the target sentences are not reversed). By doing so, the LSTM’s test perplexity dropped from 5.8 to 4.7, and the test BLEU scores of its decoded translations increased from 25.9 to 30.6.

While we do not have a complete explanation to this phenomenon, we believe that it is caused by the introduction of many short term dependencies to the dataset. Normally, when we concatenate a source sentence with a target sentence, each word in the source sentence is far from its corresponding word in the target sentence. As a result, the problem has a large “minimal time lag” [17]. By reversing the words in the source sentence, the average distance between corresponding words in the source and target language is unchanged. However, the first few words in the source language are now very close to the first few words in the target language, so the problem’s minimal time lag is greatly reduced. Thus, backpropagation has an easier time “establishing communication” between the source sentence and the target sentence, which in turn results in substantially improved overall performance.

Figure from https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html

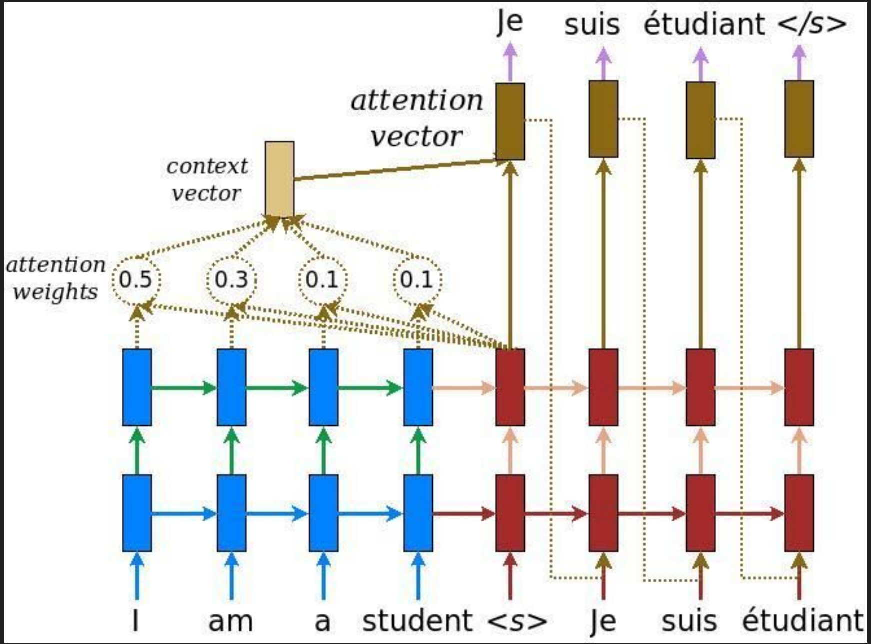
Attention Mechanism (2014)

Enables "adaptive context vector" for each decoding timestep

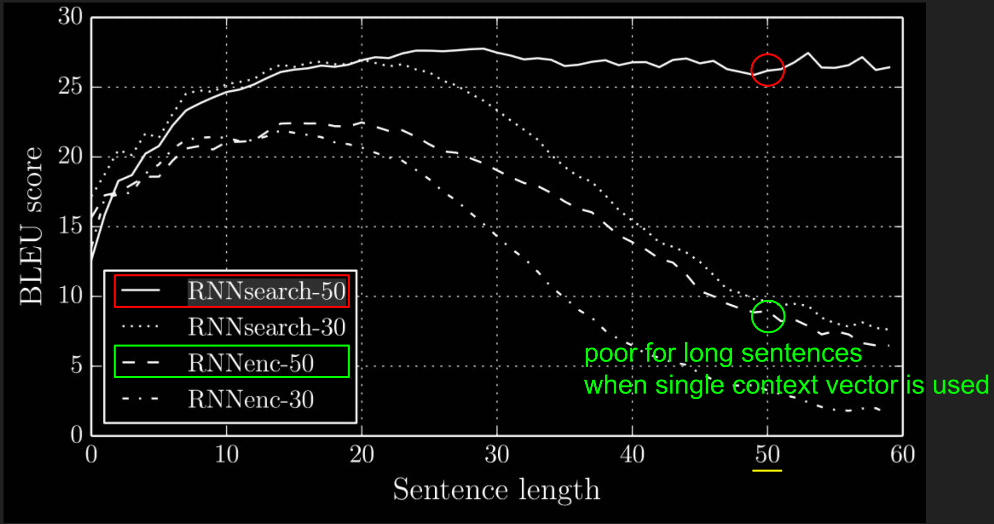
Query (Q): Target sequence state
Key (K): Source sequence state
Value (V): A state corresponding to the key

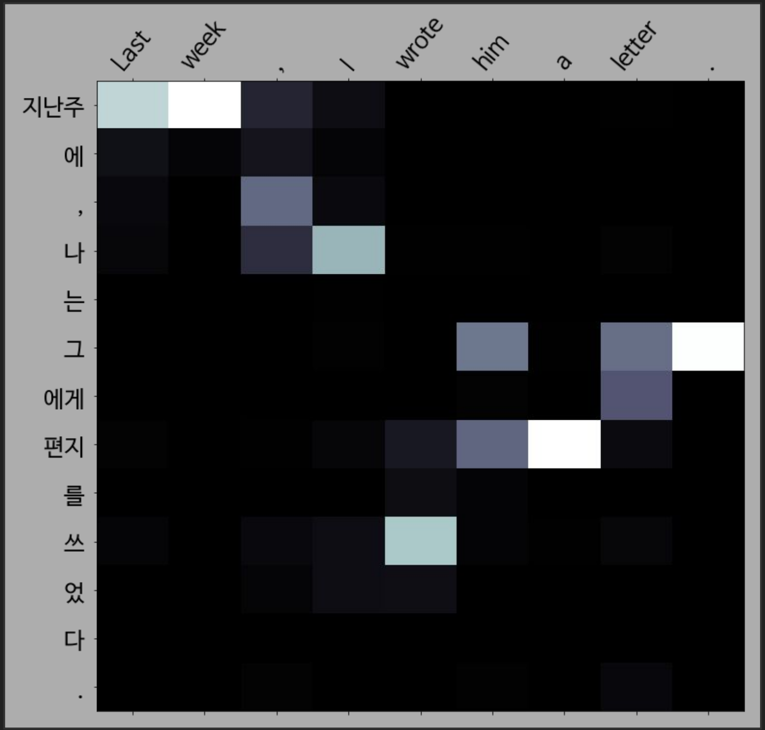
For each decoder timestep:
we do a weighted sum of V_i s
with scalar score from Q and K_i s

- K=V for many cases
- V is often referred as "memory"



NMT with Attention works well with *long* sentences! (**)





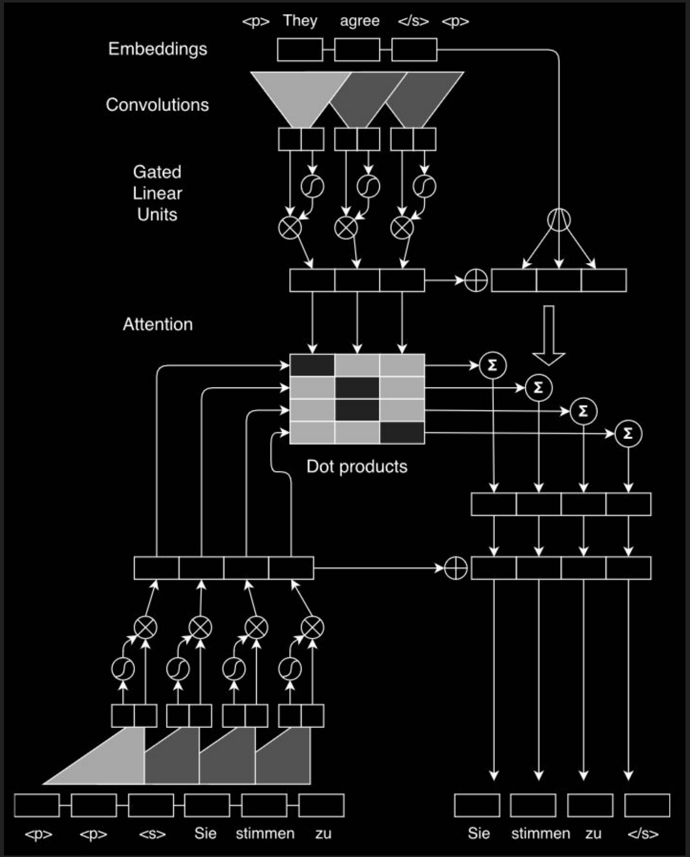
(*) "long-term" in old days: 50 steps
(**) long data also required for training

Figure from https://github.com/jaywalnut310/Attention-Mechanisms

Limitation of vanilla attention mechanism

Great at capturing source-target dependency, but:

- source-source dependency?
- target-target dependency?



"Conv seq2seq": convolution + attention (May 2017, FAIR)

Attention Is All You Need

Ashish Vaswani*
Google Brain
avaswani@google.com

Noam Shazeer*
Google Brain
noam@google.com

Niki Parmar*
Google Research
nikip@google.com

Jakob Uszkoreit*
Google Research
usz@google.com

Llion Jones*
Google Research
llion@google.com

Aidan N. Gomez* †
University of Toronto
aidan@cs.toronto.edu

Łukasz Kaiser*
Google Brain
lukaszkaiser@google.com

Illia Polosukhin* ‡
illia.polosukhin@gmail.com

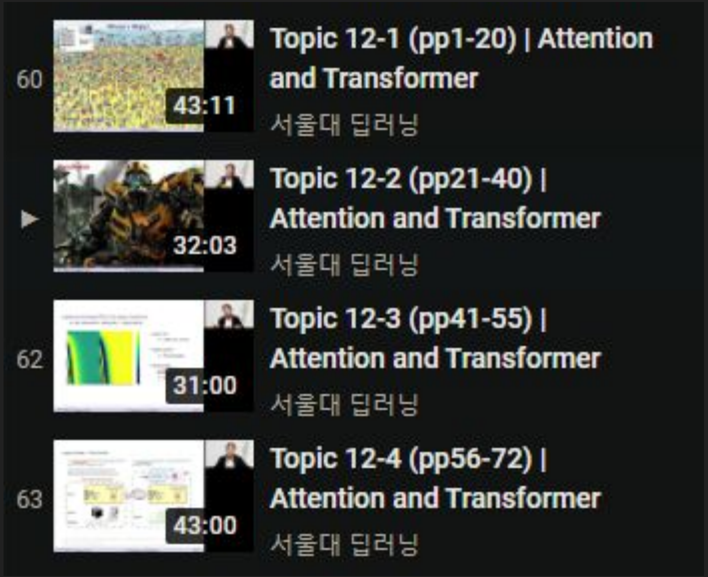
Works that made Deep Learning explode in 2010s

- AlexNet	(2012, cited by 93089)	
- Dropout	(2014, cited by 33155)	
- GAN	(2014, cited by 39297)	
- Adam optimizer	(2014, cited by 94782)	
- Batch normalization	(2015, cited by 32985)	
- Residual connection	(2015, cited by 102678)	
- Transformer	(2017, cited by 34018)	<- we're here
- BERT	(2018, cited by 32208)	

… and powerful GPUs

Details of Transformer (which is too much)

- Attention
  - Multi-Head Self Attention
  - Multi-Head Attention
  - Masked Multi-Head Self Attention
- Positional Encoding: absolute / relative
- Residual connection
- Weight Tying
- Label smoothing
- Decoding strategy: greedy / beam-search
- Byte-pair encoding
- Learning rate scheduling (warmup)



Refer to https://youtu.be/MpWxzzADroA if you're interested in comprehensive details of the Transformer ("윤성로 딥러닝" @ YouTube)

Overall

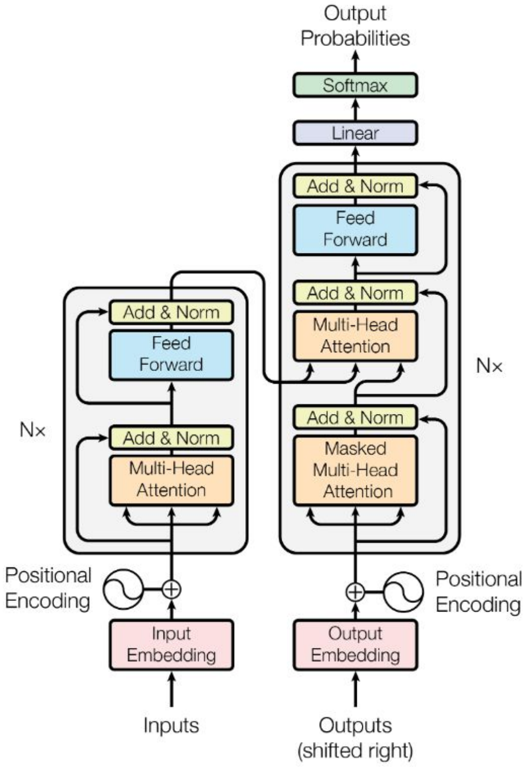


Figure 1: The Transformer - model architecture.

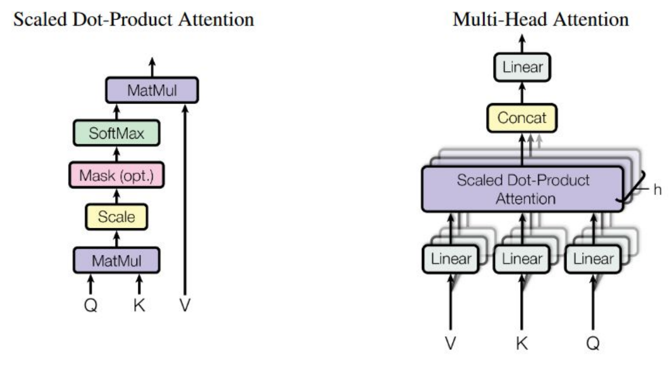
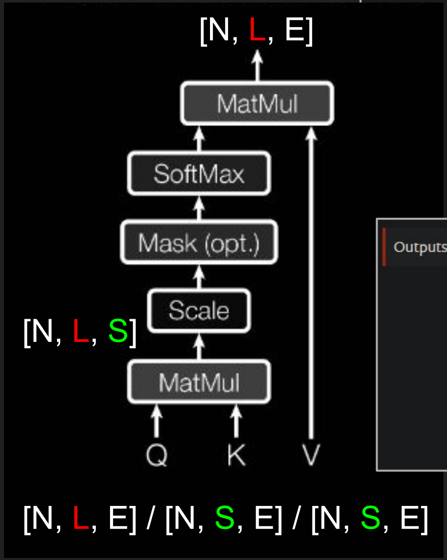


Table 2: The Transformer achieves better BLEU scores than previous state-of-the-art models on the English-to-German and English-to-French newstest2014 tests at a fraction of the training cost.

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	$\mathbf{3.3 \cdot 10^{18}}$	
Transformer (big)	**28.4**	**41.8**	$2.3 \cdot 10^{19}$	

Multi-Head (Self) Attention



`forward(query, key, value, key_padding_mask=None, need_weights=True, attn_mask=None)` [SOURCE]

Parameters (from torch.nn.MultiheadAttention)

- **query** – Query embeddings of shape (L, N, E_q) when `batch_first=False` or (N, L, E_q) when `batch_first=True`, where L is the target sequence length, N is the batch size, and E_q is the query embedding dimension `embed_dim`. Queries are compared against key-value pairs to produce the output. See "Attention Is All You Need" for more details.
- **key** – Key embeddings of shape (S, N, E_k) when `batch_first=False` or (N, S, E_k) when `batch_first=True`, where S is the source sequence length, N is the batch size, and E_k is the key embedding dimension `kdim`. See "Attention Is All You Need" for more details.
- **value** – Value embeddings of shape (S, N, E_v) when `batch_first=False` or (N, S, E_v) when `batch_first=True`, where S is the source sequence length, N is the batch size, and E_v is the value embedding dimension `vdim`. See "Attention Is All You Need" for more details.

Outputs:

- **attn_output** - Attention outputs of shape (L, N, E) when `batch_first=False` or (N, L, E) when `batch_first=True`, where L is the target sequence length, N is the batch size, and E is the embedding dimension `embed_dim`.
- **attn_output_weights** - Attention output weights of shape (N, L, S), where N is the batch size, L is the target sequence length, and S is the source sequence length. Only returned when `need_weights=True`.

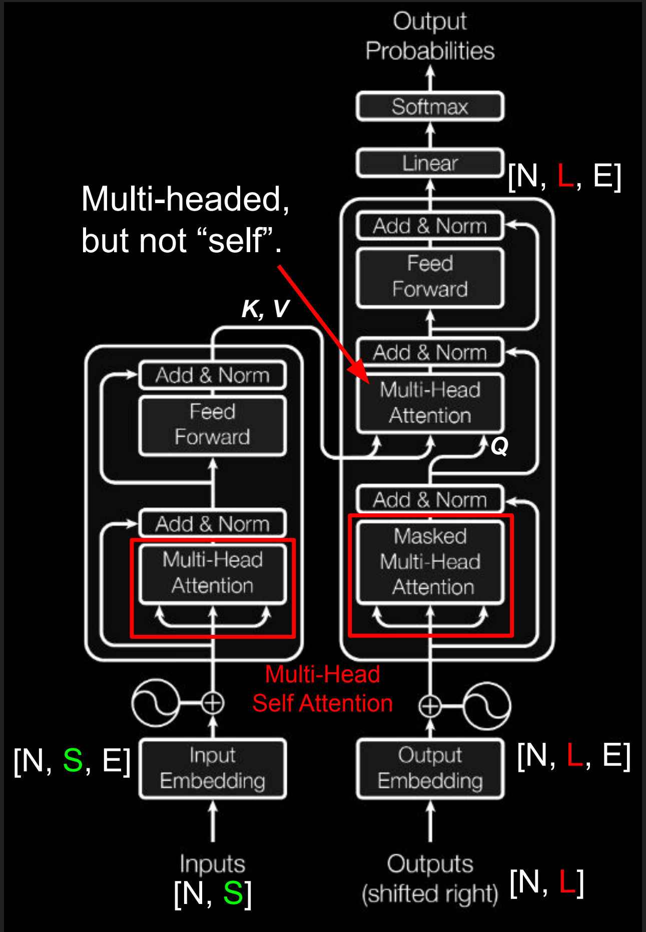
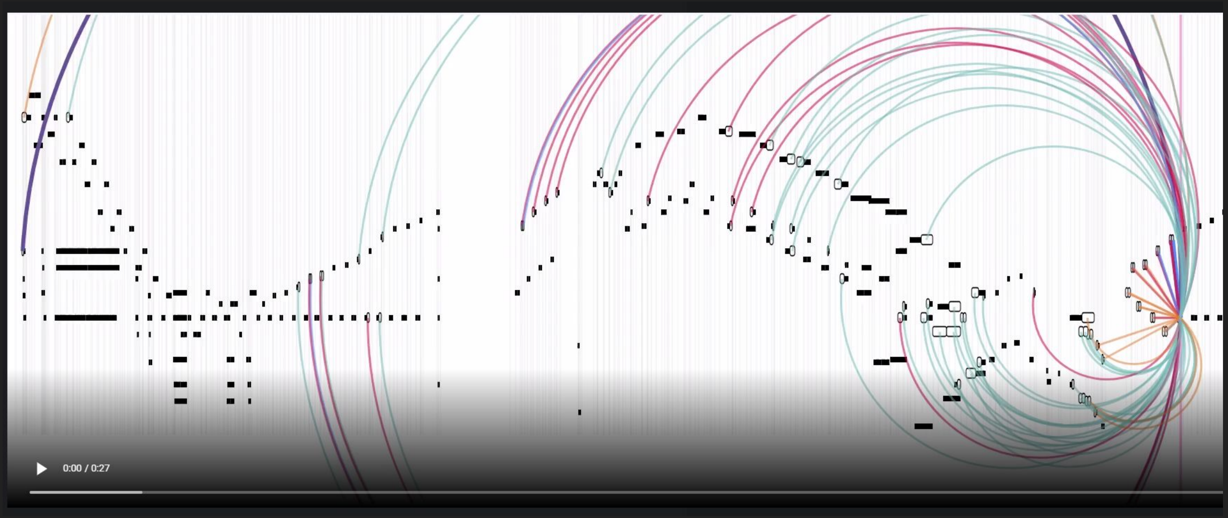


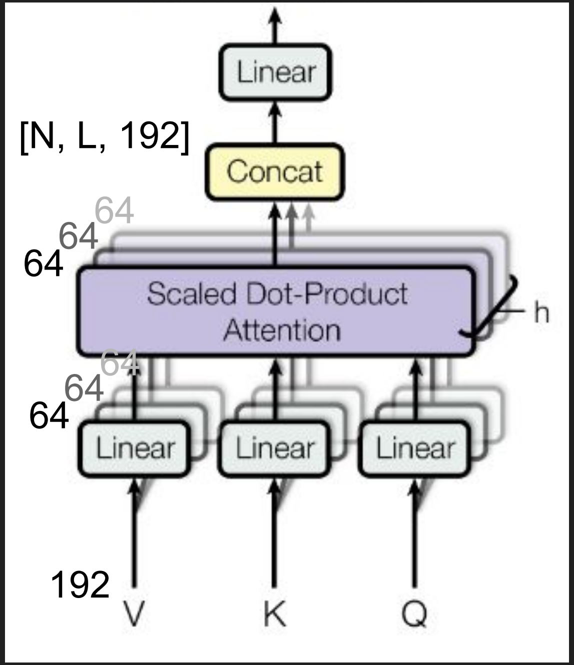
Figure 1: The Transformer - model architecture.

Refer to https://pytorch.org/docs/stable/generated/torch.nn.MultiheadAttention.html

Why multi-head?

Allows to jointly attend to info. from different subspaces





Screenshot from video at https://magenta.github.io/music-transformer-visualization/

Why self-attention?

1) Computational complexity per layer
2) Parallelizable computation

    no need for sequential operation across timesteps

3) Path length between long-range dependency

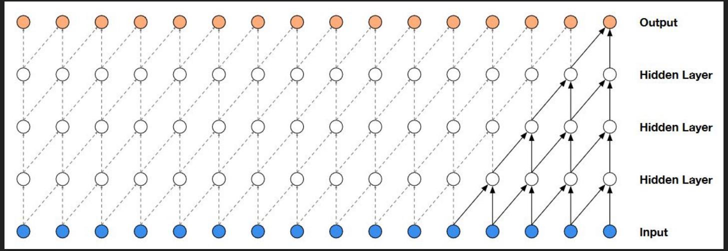
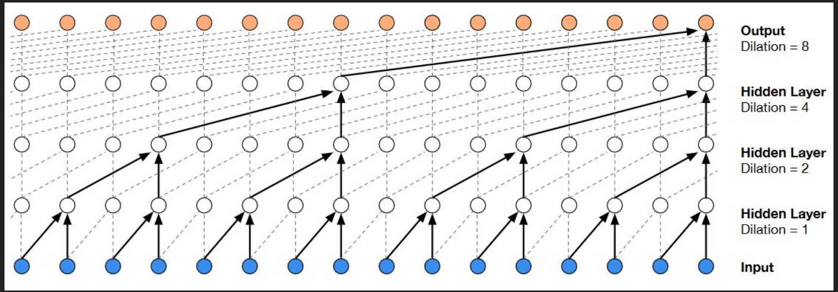


Table 1: Maximum path lengths, per-layer complexity and minimum number of sequential operations for different layer types. n is the sequence length, d is the representation dimension, k is the kernel size of convolutions and r the size of the neighborhood in restricted self-attention.

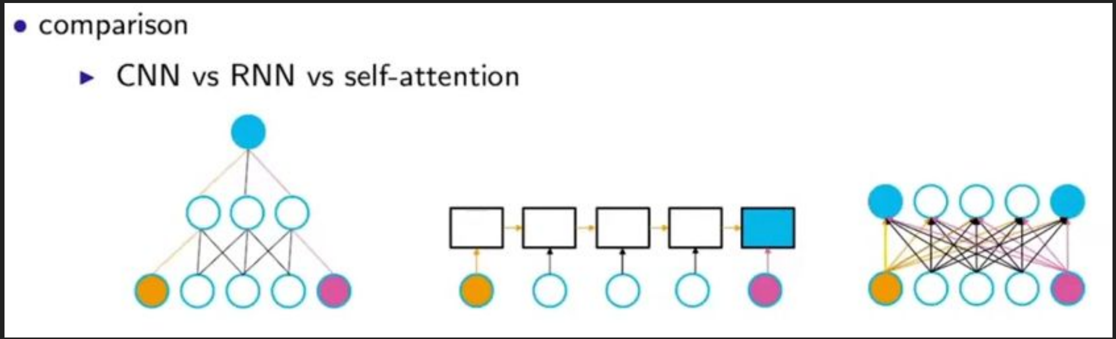
Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

Dilated convolution of WaveNet ->



More on "Path length"

… or "receptive field"



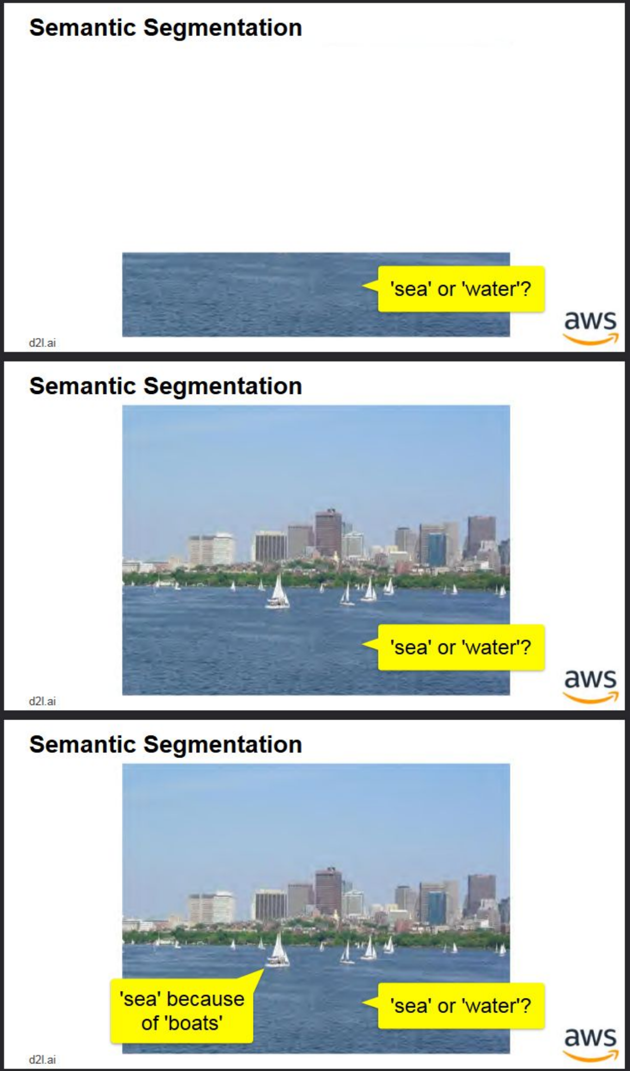


Figure from https://youtu.be/MpWxzzADroA?t=2090
Figure from https://icml.cc/Conferences/2019/ScheduleMultitrack?event=4343

Position Encoding

- Multi-Head Self Attention is permutation *equivariant*
  - if, input (1, 2, 3) -> output (a, b, c)
  - then input (1, 3, 2) -> output (a, c, b)
  - equivariance *vs.* invariance
- We must provide some position information of each input tokens
  - without pos. enc., “Alice hit Bob” will be same with “Bob hit Alice”
- Traditional CNN/RNNs didn’t require position encoding
  - … but might benefit from them!
  - I wrote a blog post on position encoding; see link below if you’re interested!

https://blog-deepest.medium.com/position-encoding%EC%9D%98-%EC%A2%85%EB%A5%98%EC%99%80-%EB%B6%84%EC%84%9D-ab1816b0f62a (written in Korean, “Position Encoding의 종류와 분석”)

Common misconceptions of Transformer

All of the following statements are false:

- Multi-Head attention is always a self-attention?
  - No. Both are extensively utilized in transformer, but cross-attention can be also multi-head.
- Transformer generates sequence in parallel?
  - No. Training transformer works without recurrence, but not for inference (generation) phase.
- Receptive field of transformer is infinite?
  - No. Though LMs like GPT-3 can generate infinitely long sequence,
    they can only refer to *MAX_LEN* number of (previous) tokens.
  - Size of attention map is always (MAX_LEN * MAX_LEN), regardless of the input length.
  - Transformer-XL alleviates this problem

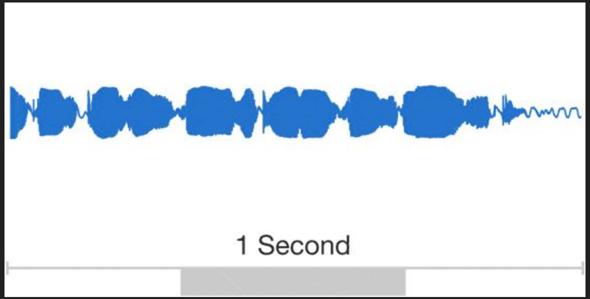
Aside: Why don't we directly generate raw audio with Trfm?

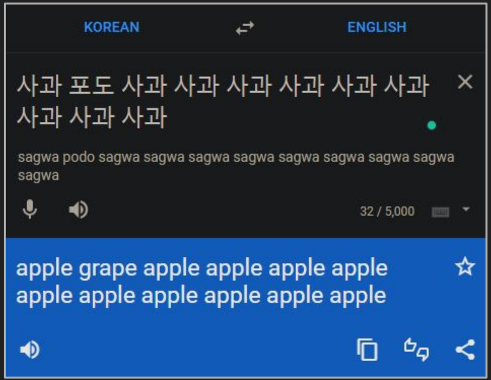
(Question from Jinhyung at last week)

My answer:

(1) Each elements from the raw audio contain too small amount of information
(2) Transformers are (known to be) fragile against highly repetitive data
(3) Looooooong-term dependency (65536*65536 attention map = 16GB) (*)

-> Hierarchical modeling might help (e.g. Jukebox, VQGAN)





(*) float32 = 4Byte; this doesn't even count Adam optimizer's EMA.

Self-supervised learning

Brief taxonomy of Deep Learning

Supervised / Semi-supervised / Unsupervised learning

- Self-supervised learning: One branch of unsupervised learning
- ‘Unsupervised’ is too broad, and could be misleading;
  Please use ‘self-supervised’ as possible

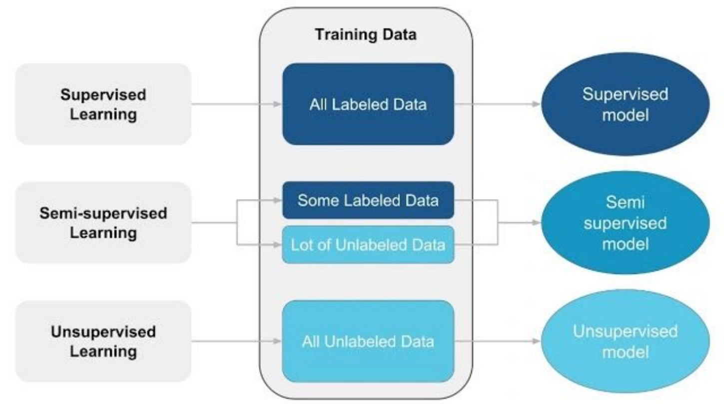
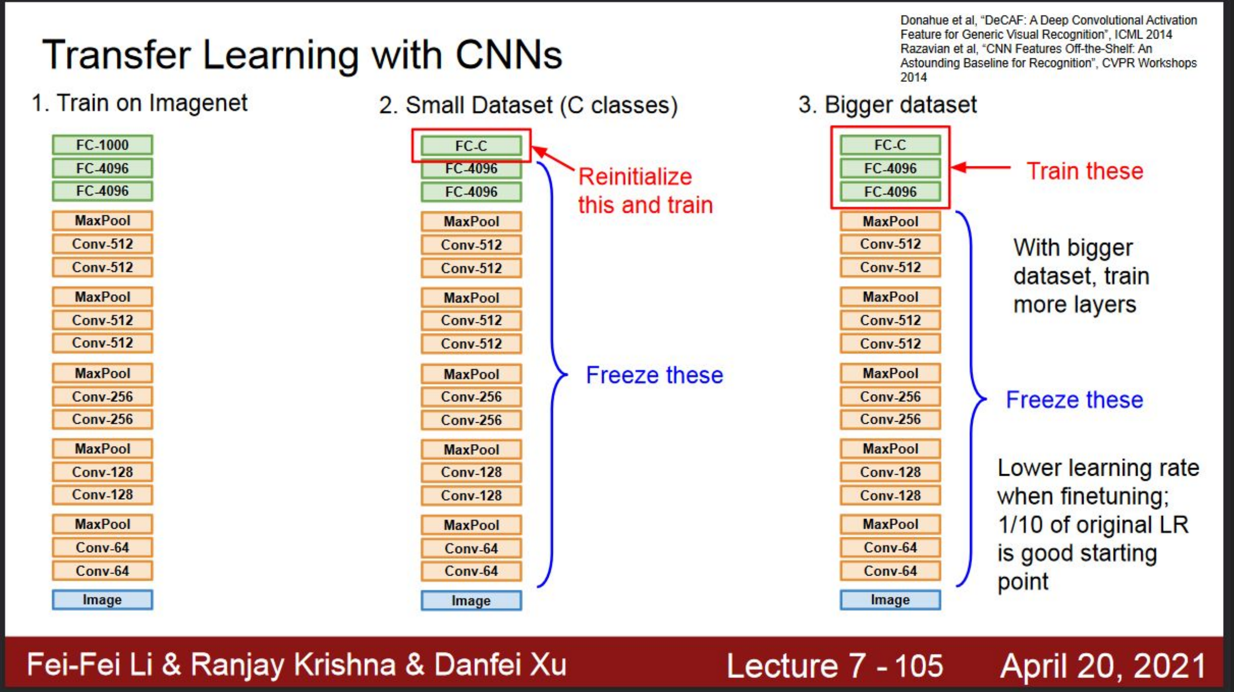


Figure from https://business.blogthinkbig.com/semi-supervised-learning-the-great-unknown/

Pre-train the model and transfer (or “fine-tune”)

~~“You need a lot of a data if you want to train/use CNNs”~~ not anymore!

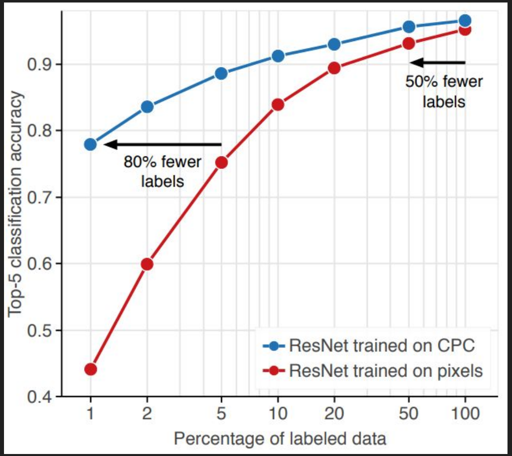


Slides from http://cs231n.stanford.edu/slides/2021/lecture_7.pdf

Why self-supervised learning matters?

(It's basically self-supervised pre-trained representations)

- Better data efficiency
  - High-quality labelled data are very expensive $$$
    (Caveat: collecting unlabelled data also costs $)
- Higher data availability
  - So many unlabelled data available on the Internet
    #(images): JFT-300M >> ImageNet (1M)
    #(words): en-Wikipedia (2.5B) >> SQuAD (100k Q&A pairs)
- Wider transfer capability
  - Might even work with a new task in zero-shot manner



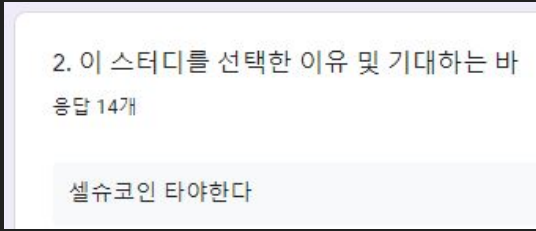
CPC v2 (arXiv:1905.09272)

Why self-supervised learning matters?

"The next AI revolution will not be supervised or purely reinforced. The future is self-supervised learning with massive amounts of data and very large networks."

- Yann LeCun, 2019

Inventor of LeNet (1989), the first CNN-based digit classifier trained with backprop
Turing award co-recipient (2018)
VP & Chief AI Scientist @ Meta (Facebook)

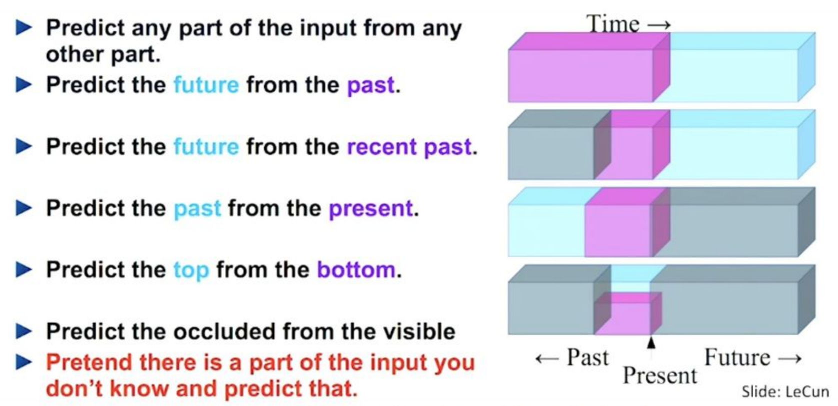


Examples of pretext tasks

How to make a “label” with the data itself?

- Generative modelling
  - Autoregressive Generation (e.g. GPT = Generative Pre-Training)
  - Masked Generation (e.g. BERT)
  - Colorization, Super resolution, …

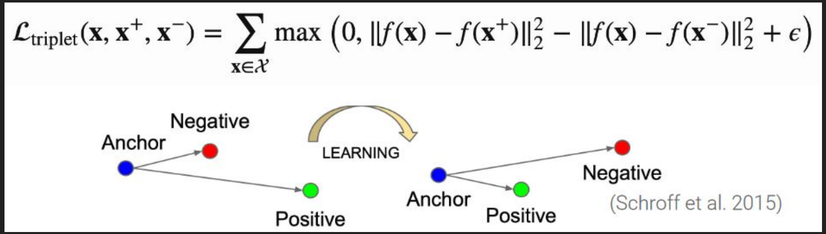
- Innate Relationship Prediction
  - Predict order of image patches, rotation, …



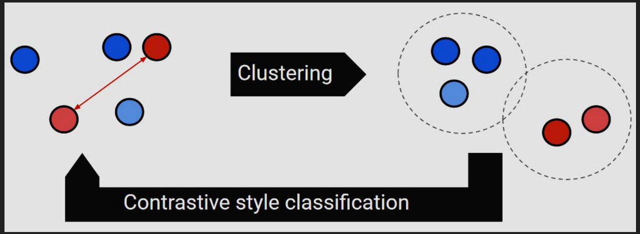
(Famous illustration from Yann LeCun)

Examples of pretext tasks (cont'd)

- Contrastive Learning
  - Inter-sample classification
    (e.g. Triplet loss, a.k.a "metric learning")

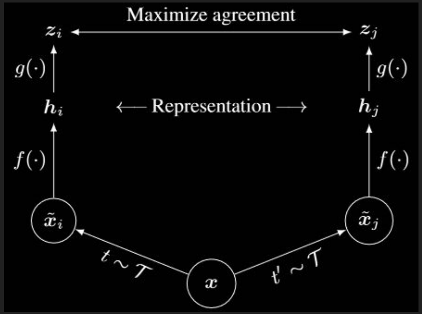


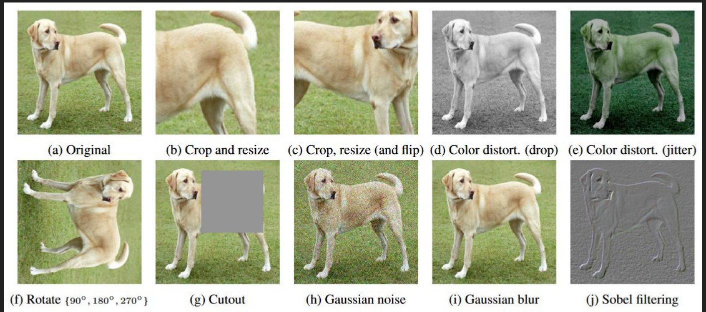
  - Feature clustering (e.g. HuBERT)
    generate pseudo-labels with clustering algorithms



  - Multiview coding (e.g. CMC, SimCLR, …)
    based on classic hypothesis:
    "powerful representation is one that models view-invariant factors"

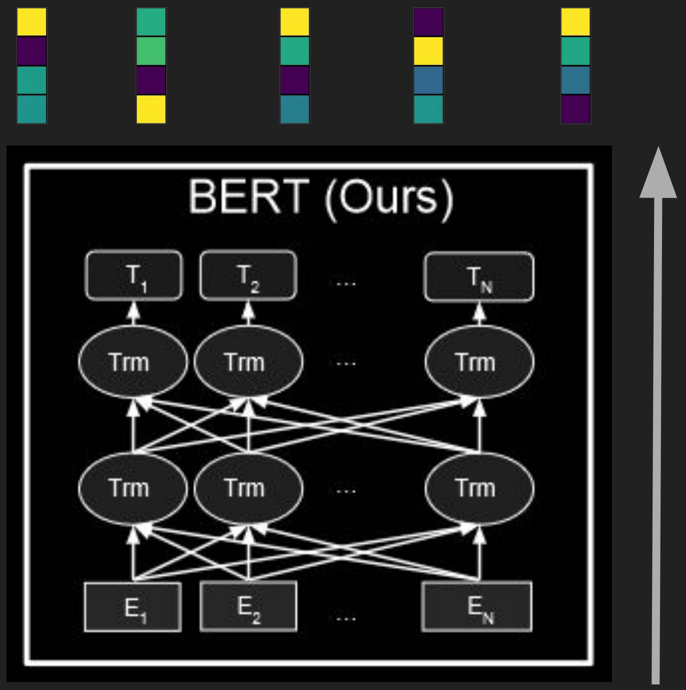
SimCLR (arXiv:2002.05709)





BERT: Self-supervised learning meets Transformer

Bidirectional
Encoder
Representations from
Transformers

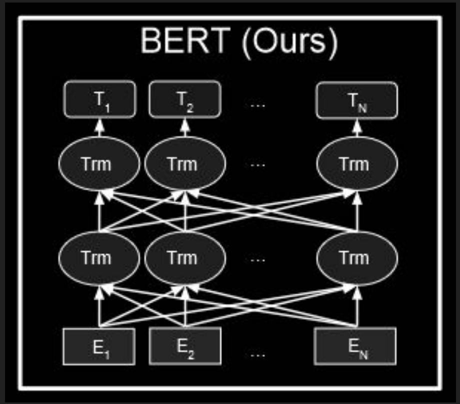


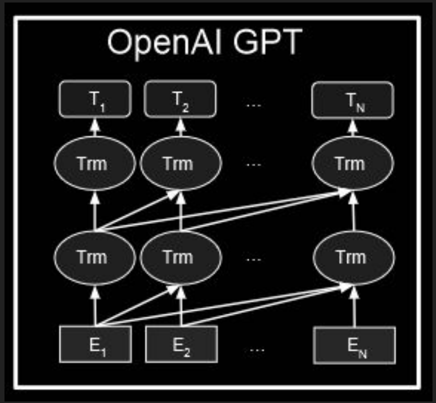
[CLS] / “배” / “먹고” / “배” / “아프다”

[CLS] token is interchangeably referred as [BOS]

Bidirectional: better "context"

- BERT: Masked LM
    - Uses only Transformer encoder (always parallelizable!)
- GPT: Autoregressive generation (Left-to-Right)
    - Uses only Transformer decoder
      (GPT's main focus is not representation learning anymore)





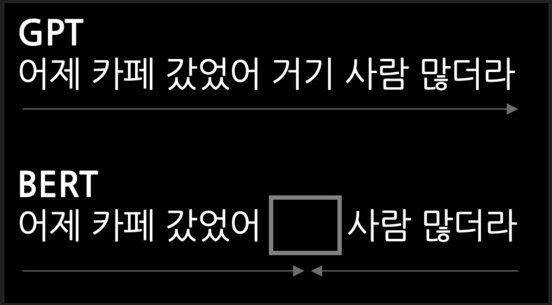


Figure from https://ratsgo.github.io/nlpbook/docs/language_model/bert_gpt/

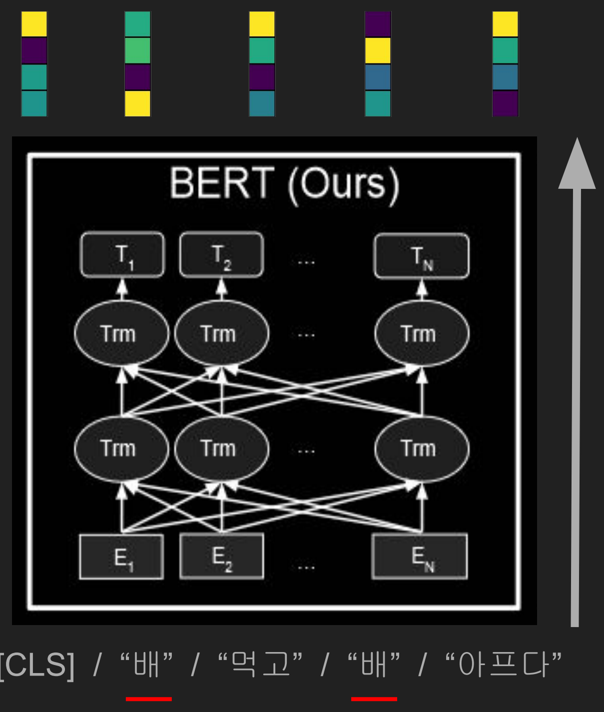
"Trm" = Transformer / LM = Language Model

Encoder **R**epresentations: beyond 'word embeddings'

word2vec: pre-trained but not contextualized
(2013, cited by 31405)

BERT works better for:

- pronouns (대명사)
- homophones (동음이의어)
- facts that depend on time

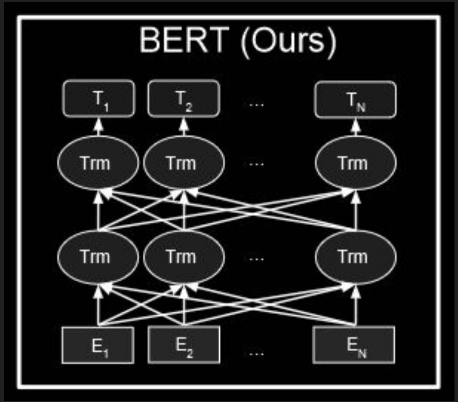


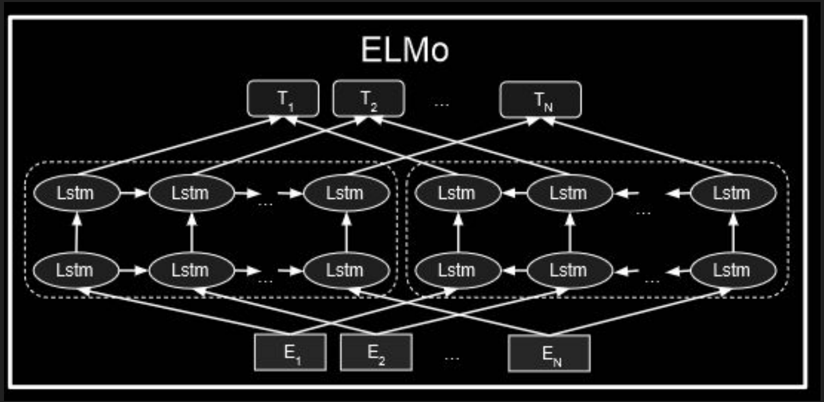
from Transformers: better NN architecture

BERT was not a first “contextualized representation” for NLP

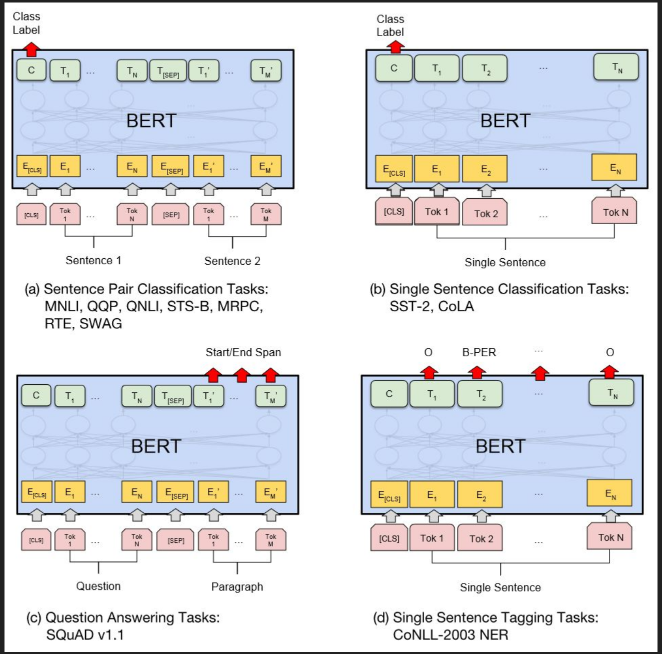
pros of Transformers, compared to LSTM:

- significantly faster calculation
  - benefit from GPU parallelism
- better path length
  - full connection across sequence
  - especially important for NLP





SotA on 11 downstream NLP tasks



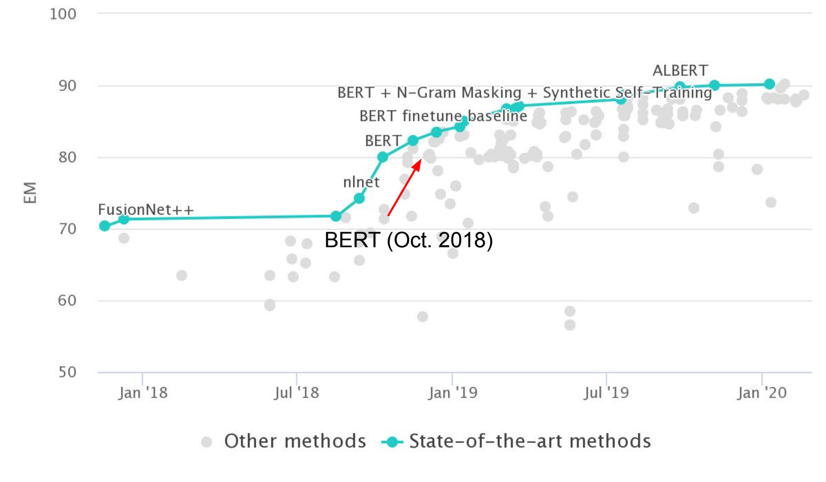


Figure (middle) from https://paperswithcode.com/sota/question-answering-on-squad20

Things made BERT so successful

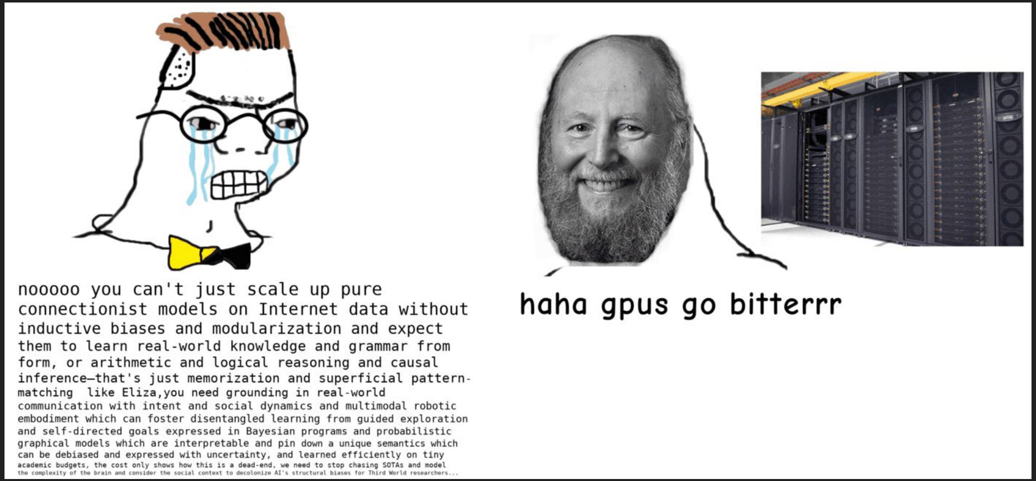
- Self-Supervised Learning
  - no need for expensive, massive labelling
- Massive amount of dataset
  - English Wikipedia (2.5B words, ~17GB in plain text)
- Architecture choice: Transformer
  - massive gain on both training speed and performance

So, this leads to an important question: “Does it scale?”

The Bitter Lesson (Sutton, 2019)

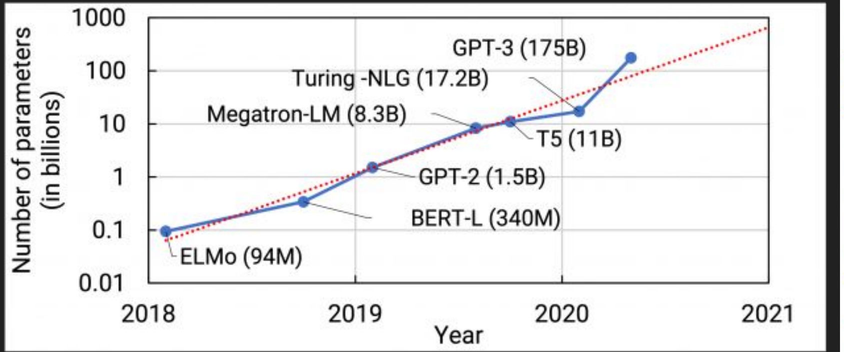
“Most AI research has been conducted as if the computation available to the agent were constant … ”

“Deep learning methods rely even less on human knowledge, and use even more computation, together with learning on huge training sets, to produce dramatically better speech recognition systems.”



http://www.incompleteideas.net/IncIdeas/BitterLesson.html

GPU/Data resources that are out of reach …



4.1 PRETRAINING

We use the wav2vec 2.0 implementation available in fairseq (Ott et al., 2019) and evaluate several model architectures detailed in Table 2. We consider models with between 0.3B parameters to 2B parameters. To optimize GPU memory usage, we use a fully sharded backend (Rajbhandari et al., 2021) as well as activation checkpointing (Chen et al., 2016) as implemented in FairScale (Baines et al., 2021).

Models are optimized with Adam (Kingma & Ba, 2015) and the learning rate is warmed up for the first 32K steps followed by polynomial decay to zero for the remainder of training. Training audio sequences are cropped to a maximum of 320K samples, or 20 seconds and all models were pretrained for a total of one million updates. XLS-R (0.3B) was trained on 128 GPUs with nearly 2M samples on each GPU, totaling about 4.3h of data in a batch. Larger models were trained on 200 GPUs with 800K to 1M samples on each GPU giving an effective batch size of about 2.8-3.6 hours.

Our training data covers 128 languages and five training corpora with different characteristics. To balance data from the different languages and corpora we upsample both training corpora and languages. We first upsample the languages within a particular corpus using the strategy outlined in §2 and then balance the different corpora using the same strategy by treating each corpus as a different language. We use $\alpha = 0.5$ in all cases.

XLS-R (arXiv:2111.09296)

Figure from https://developer.nvidia.com/blog/scaling-language-model-training-to-a-trillion-parameters-using-megatron/

Good news: Scalability is not only a way of our progress

- How could we make it efficient?
  - less parameters with better training strategy: ALBERT
  - less number of training steps: ELECTRA
  - less energy: MobileBERT
- How should we bring it back to smaller downstream tasks?
  - Fortunately, there are so many freely available pre-trained models
- Should we only use natural language for NLP?
  - The ultimate goal: make it multi-modal

Experience Grounds Language

arXiv:2004.10151

Yonatan Bisk* **Ari Holtzman*** **Jesse Thomason***
Jacob Andreas Yoshua Bengio Joyce Chai Mirella Lapata
Angeliki Lazaridou Jonathan May Aleksandr Nisnevich Nicolas Pinto Joseph Turian

You can't learn language …

… from the radio (internet).	WS2 < WS3
… from a television.	WS3 < WS4
… by yourself.	WS4 < WS5

We define five levels of **World Scope**:

WS1. Corpus *(our past)*
WS2. Internet *(our present)*
WS3. Perception
WS4. Embodiment
WS5. Social

Conclusion with Recap Questions

Recap questions

- Does self-supervised learning completely eliminates the need of labeled data? (answer: no, we often need them for downstream tasks)
- What makes self-supervised learning difficult (for us)?
- Why self-supervised learning is so popular for model pre-training?
- What was a main motivation of BERT?
- Describe at least 3 different pretext tasks for self-supervision.
- Which is which? (BERT/GPT) uses only Transformer (Encoder/Decoder).
- What makes transformer better than vanilla attention (for NMT?)
- Can you draw the Transformer architecture without looking at the paper? (?)

Useful links

- Lilian Weng & Jong Wook Kim, Self supervised Learning, NeurIPS 2021 Tutorial
  - https://nips.cc/media/neurips-2021/Slides/21895.pdf
  - Lilian Weng's blog is great!
    https://lilianweng.github.io/lil-log/2019/11/10/self-supervised-learning.html
    https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html
- Alex Smola & Aston Zhang, Attention in Deep Learning, ICML 2019 Tutorial
  - https://icml.cc/media/icml-2019/Slides/4343.pdf

References (that were not cited in the slides)

[1, BERT] Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).

[2] Graves, Alex. "Generating sequences with recurrent neural networks." arXiv preprint arXiv:1308.0850 (2013).

[3, seq2seq] Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." Advances in neural information processing systems. 2014.

[4, vanilla attention] Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." arXiv preprint arXiv:1409.0473 (2014).

[5, convs2s] Gehring, Jonas, et al. "Convolutional sequence to sequence learning." International Conference on Machine Learning. PMLR, 2017.

[6, transformer] Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems. 2017.

[7] Huang, Anna, et al. "Visualizing Music Self-Attention." (2018).

[8, wavenet] Oord, Aaron van den, et al. "Wavenet: A generative model for raw audio." arXiv preprint arXiv:1609.03499 (2016).

[9, triplet loss] Schroff, Florian, Dmitry Kalenichenko, and James Philbin. "Facenet: A unified embedding for face recognition and clustering." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

References (cont’d)

[10, HuBERT] Hsu, Wei-Ning, et al. "HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units." arXiv preprint arXiv:2106.07447 (2021).

[11, CMC] Tian, Yonglong, Dilip Krishnan, and Phillip Isola. "Contrastive multiview coding." Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16. Springer International Publishing, 2020.

[12, SimCLR] Chen, Ting, et al. "A simple framework for contrastive learning of visual representations." International conference on machine learning. PMLR, 2020.

[13, GPT] Radford, Alec, et al. "Improving language understanding by generative pre-training." (2018).

[14, word2vec] Mikolov, Tomas, et al. "Distributed representations of words and phrases and their compositionality." Advances in neural information processing systems. 2013.

[15, ELMo] Peters, Matthew E. et al. “Deep Contextualized Word Representations.” NAACL (2018).

[16, ALBERT] Lan, Zhenzhong, et al. "Albert: A lite bert for self-supervised learning of language representations." arXiv preprint arXiv:1909.11942 (2019).

[17, ELECTRA] Clark, Kevin, et al. "Electra: Pre-training text encoders as discriminators rather than generators." arXiv preprint arXiv:2003.10555 (2020).

[18, MobileBERT] Sun, Zhiqing, et al. "Mobilebert: a compact task-agnostic bert for resource-limited devices." arXiv preprint arXiv:2004.02984 (2020).