BERT & its family

Lecturer: Hiếu Trần

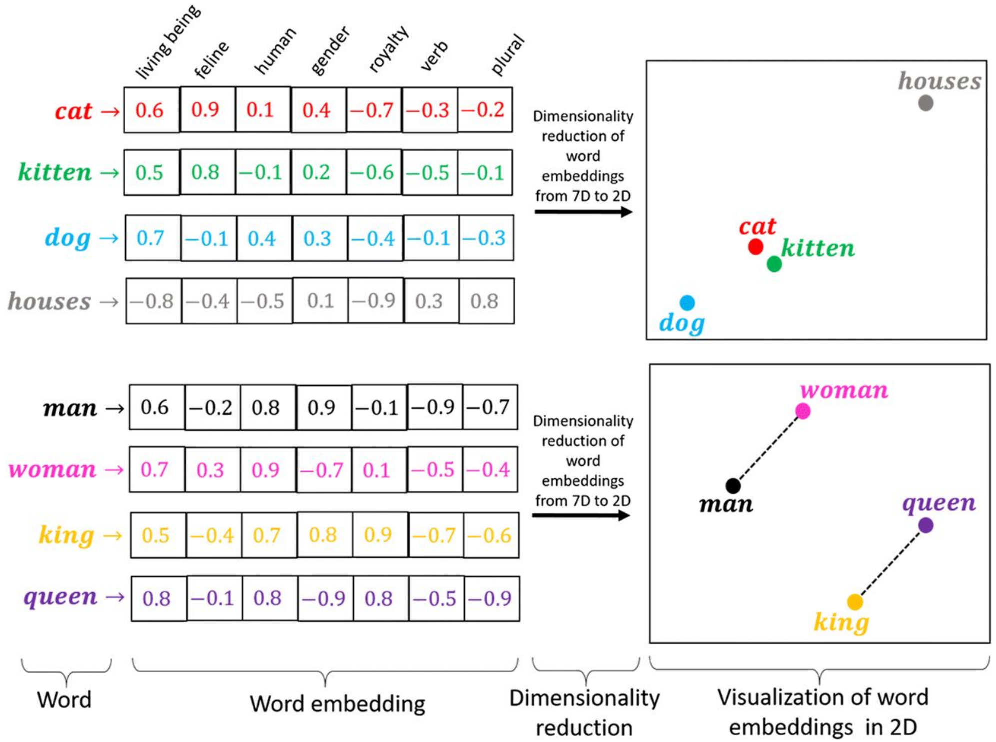
OUTLINE

- What is Contextualized Word Embedding?
- Types of Learning
- Pre-trained Language Models
- Downstream Tasks

What is Contextualized Word Embedding?

WORD EMBEDDING

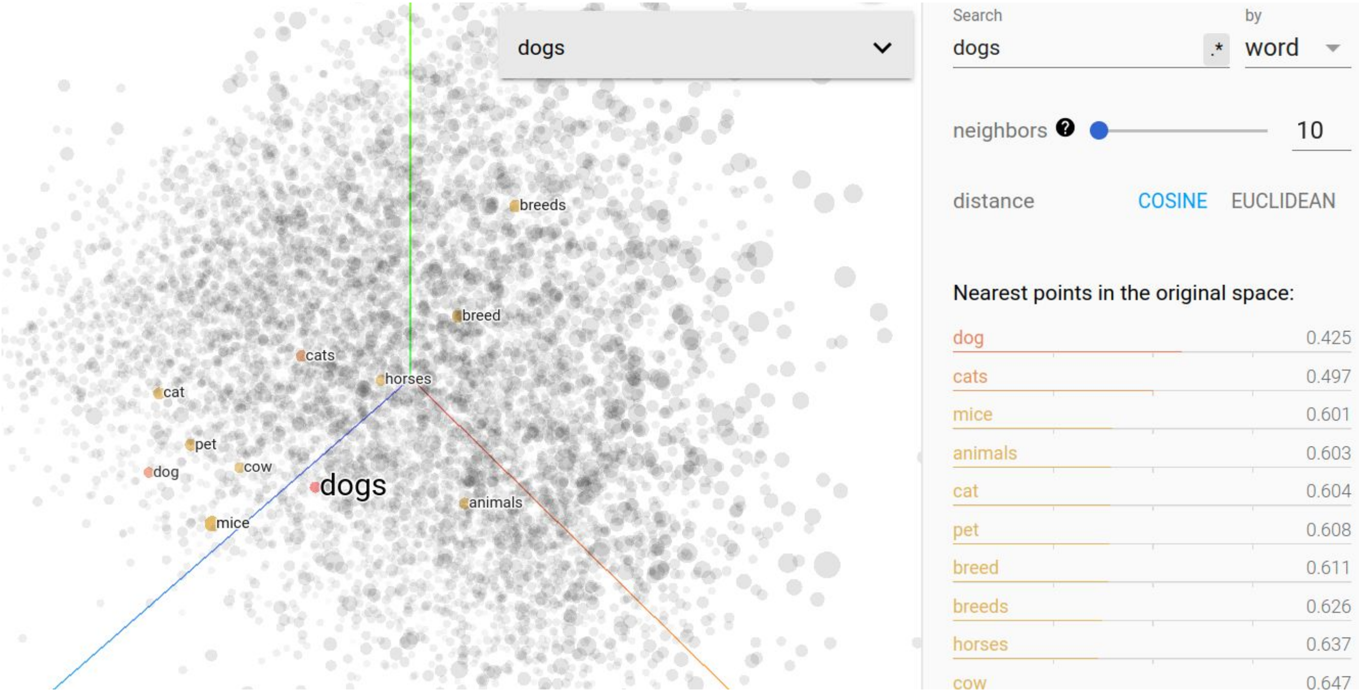
- Word embeddings are the basis of deep learning for NLP
- Word embeddings (**word2vec**, **GloVe**) are often *pre-trained* on text corpus from co-occurrence statistics



*Source: https://miro.medium.com/max/1400/1*sAJdxEsDjsPMioHyzlN3_A.png*

WORD EMBEDDING: Examples

- Visualizing Word2Vec using Tensorflow: https://projector.tensorflow.org/



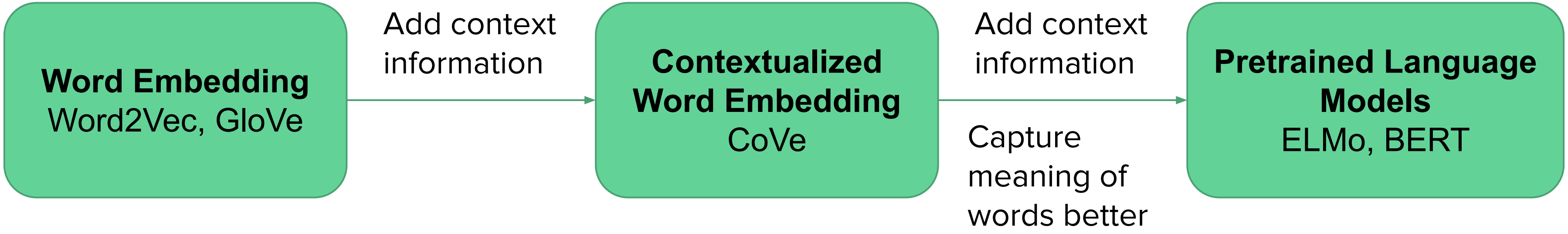
WORD EMBEDDING: Limitations

- TF-IDF: based on the frequency of words and the rarity of the words, not leverage the context i.e. word co-occurrence
- Word2Vec: based on word co-occurrences in local context
- GloVe: based on word co-occurrences in global context

Limitation: Don't have context information!!!

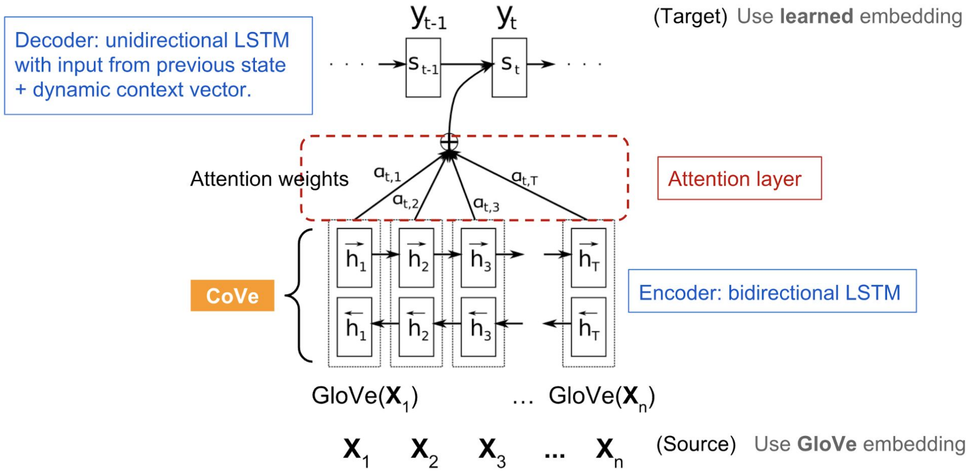
- Combine all the different meanings of the word into one vector
  - Anh ta sử dụng **kiếm** rất điêu luyện.
  - **Kiếm** ăn bây giờ khó lắm.

IMPROVEMENTS IN NLP



CONTEXTUAL WORD VECTORS (CoVe)

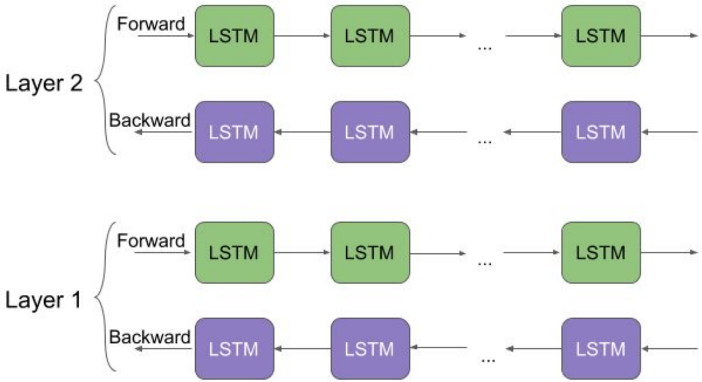
- CoVe:
  - Learn word embedding by an encoder in seq-to-seq (biLSTM) machine translation model
  - Use GloVe as an initial word embedding
- Need supervised data



Source: *https://lilianweng.github.io/posts/2019-01-31-lm/*

EMBEDDINGS FROM LANGUAGE MODEL (ELMo)

- ELMo:
  - Predict a token based on the history
  - Concatenation of right-to-left and left-to-right LSTMs



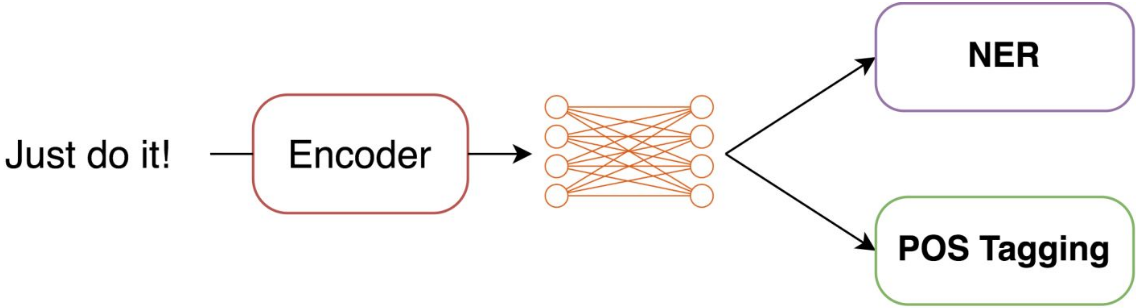
Source: https://www.analyticsvidhya.com/blog/2019/03/learn-to-use-elmo-to-extract-features-from-text/

Types of Learning

TYPES OF LEARNING

- **Multi-task learning** is a general term for training on multiple tasks.
- **Transfer learning** is a type of multi-task learning where we only really care about one of the tasks.
- **Pre-training** is a type of transfer learning where on objective is used first.

WHAT IS MULTI-TASK LEARNING?

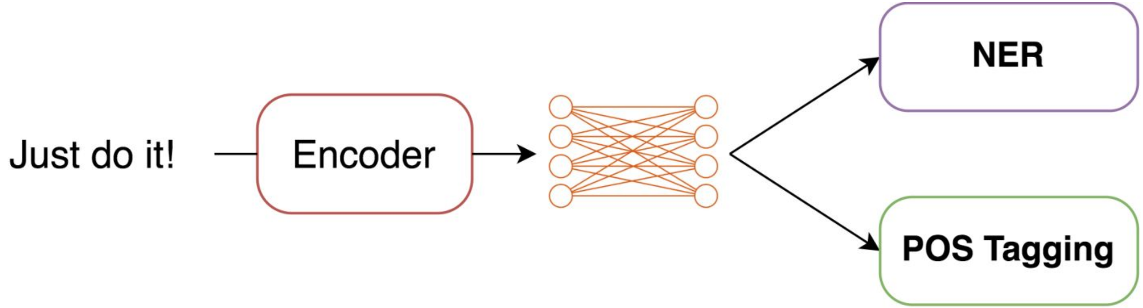


- Train representations to do well on multiple tasks at once.
- When to use multi-task?
  - When one of your tasks has many fewer data.

    e.g., high-resource language → low-resource language
  - When your tasks are related.

    e.g., predicting POS tagging and punctuation.

WHAT IS MULTI-TASK LEARNING?

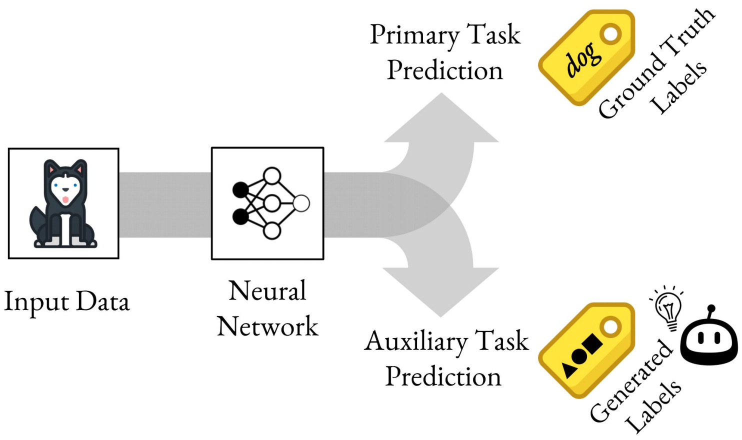


- Train representations to do well on multiple tasks at once.
- When to use multi-task?
  - When one of your tasks has many fewer data.

    e.g., high-resource language ➜ low-resource language
  - When your tasks are related.

    e.g., predicting POS tagging and punctuation.

WHEN USE MULTI-TASK LEARNING?



- When one of your tasks has many fewer data.

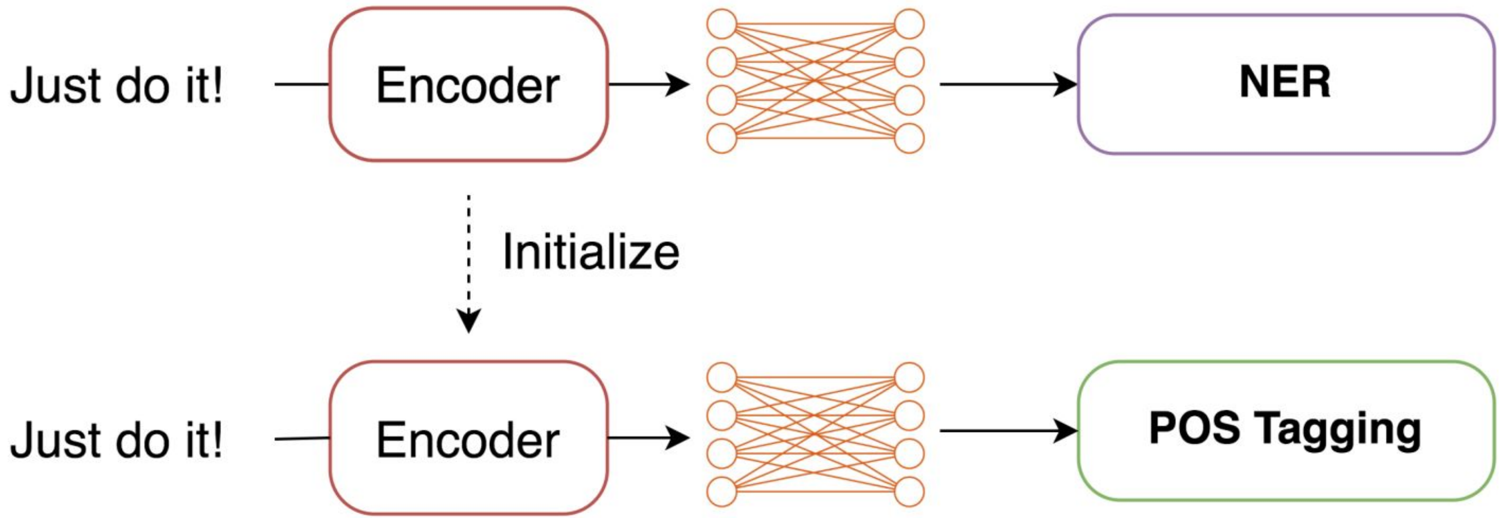
    e.g., high-resource language → low-resource language

- When your tasks are related.

    e.g., predicting POS tagging and punctuation.

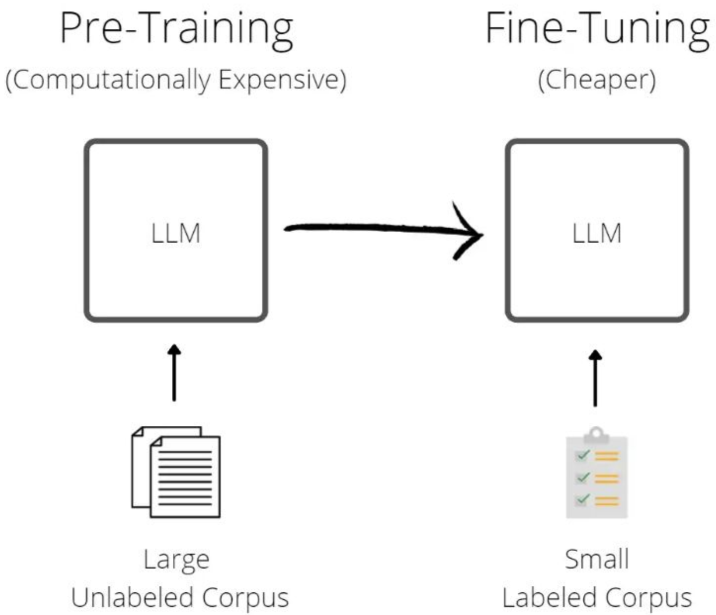
WHAT IS PRE-TRAINING?

- First train on one task, then train on another.
- Widely used in word embeddings, contextualized word representations.



WHY NEED PRE-TRAINING?

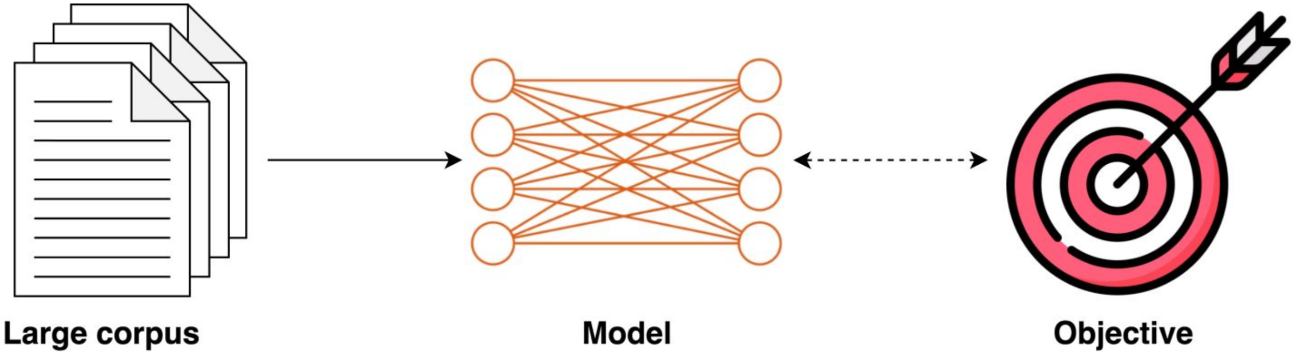
- Pre-training: train on **unlabeled data** over different pre-training objectives
  → Self-supervised pre-training
- Fine-tuning: tune parameters using **labeled data** from the downstream tasks



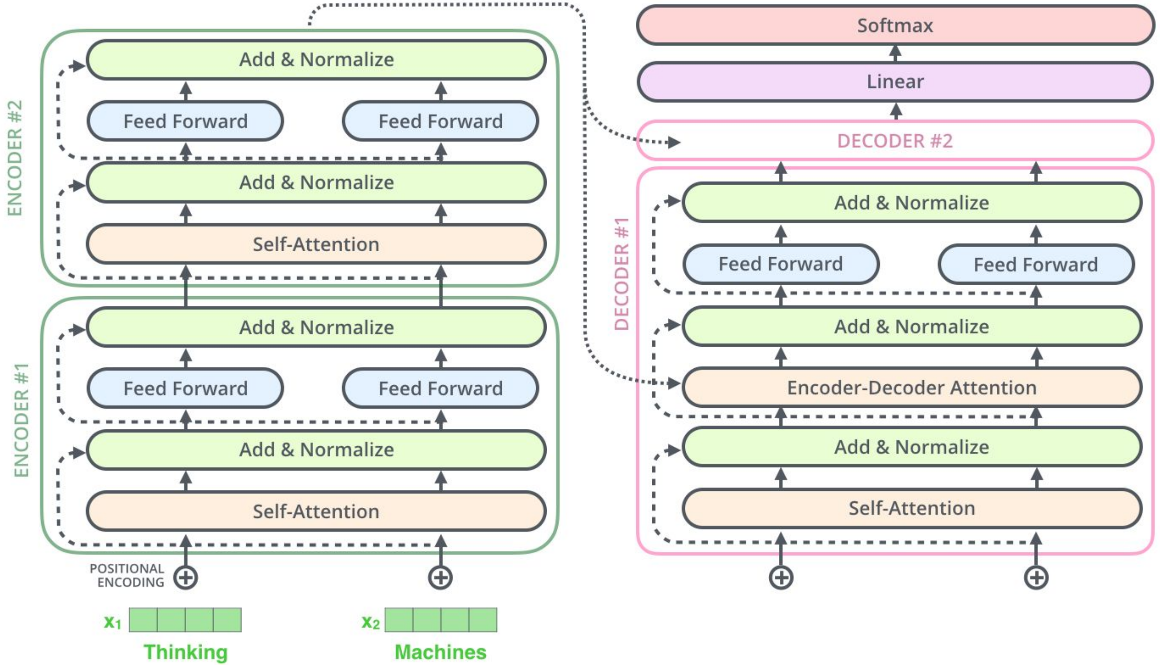
WHAT MAKE PRE-TRAINED MODELS?

Pre-training methods often refer to a combination of

- **Model**: The underlying neural network architecture
- **Training Objective**: What objective is used to pretrain
- **Data**: What data the authors chose to use to train the model



Transformer Architecture: Review



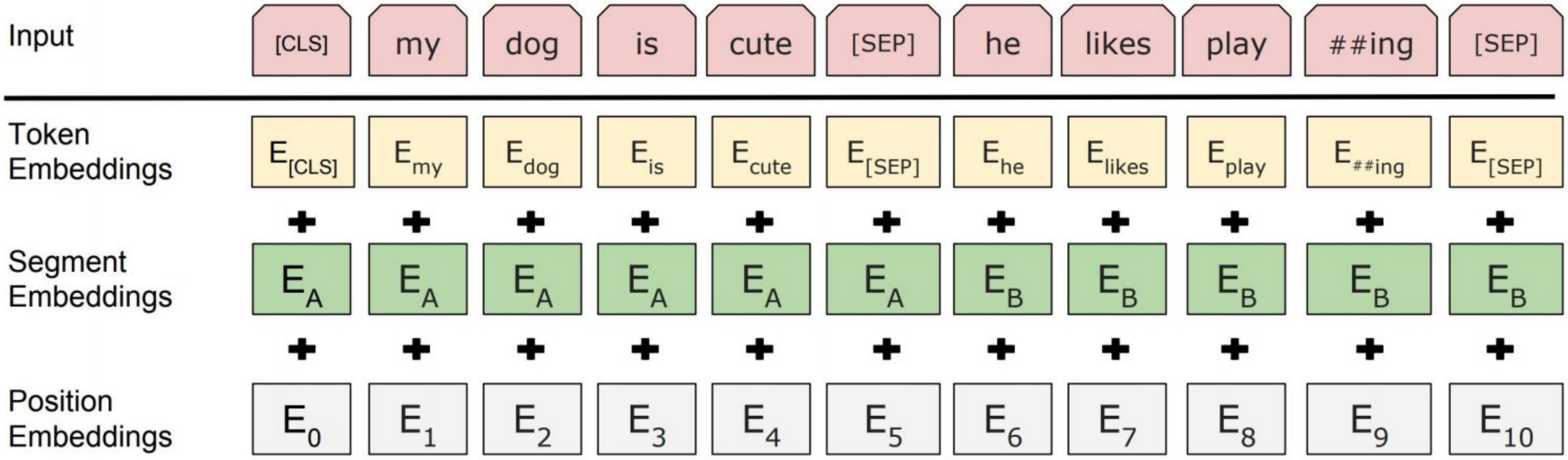
Pre-trained Language Models

BERT

- BERT - Bidirectional Encoder Representations from Transformers
- **Main idea**: Propose two pre-training objectives for bidirectional encoders
  - **MASKED LANGUAGE MODEL**
  - **NEXT SENTENCE PREDICTION**
- **Strengths**:
  - Just fine-tune BERT model for downstream tasks to outperform many heavily-engineered tasks
  - BERT advances the state of the art for eleven NLP tasks
    - question answering, textual entailment, sentence similarity

INPUT REPRESENTATION

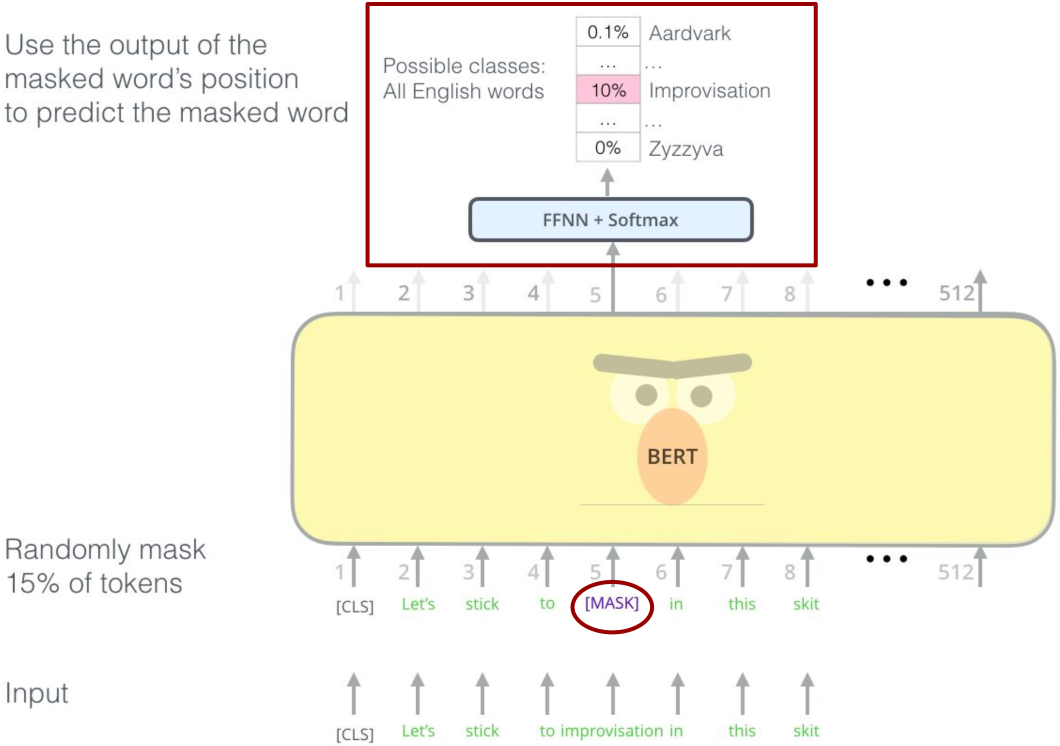
- **WordPiece** Tokenizer w/ [CLS] token, subword representation



Source: https://arxiv.org/pdf/1810.04805.pdf

MASKED LANGUAGE MODEL

- **Idea**: **Predict** the original word of a **masked word** based only on its context



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

MASKING STRATEGY

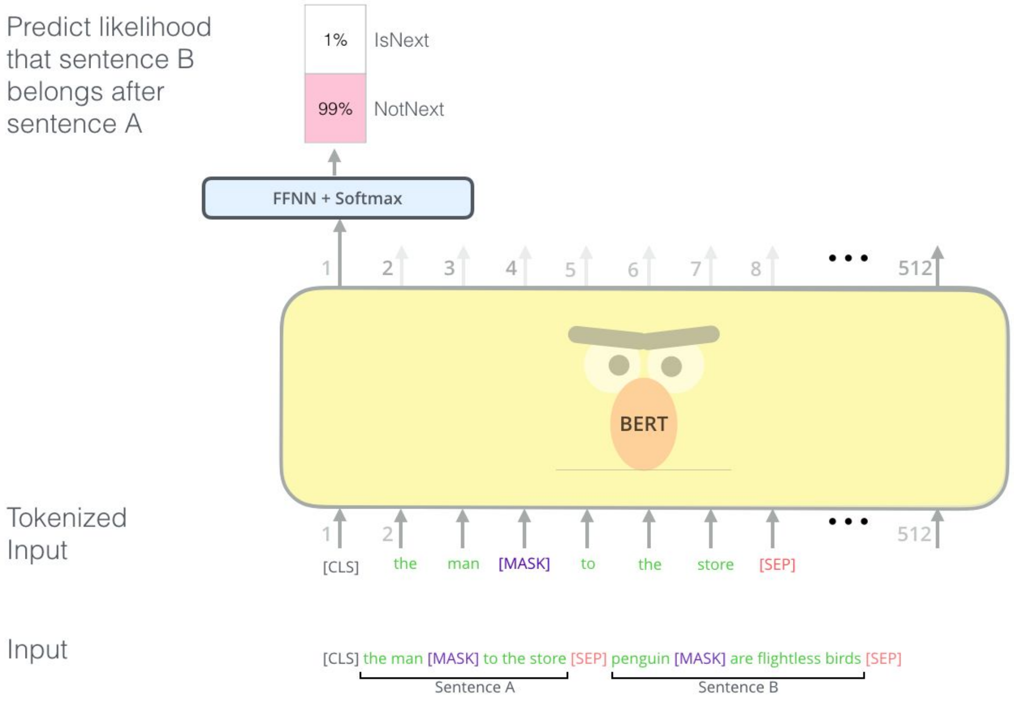
- 15% of the words are masked at random
  - Tôi muốn đến **[cửa hàng]** để mua một **[thùng]** sữa
  - Tôi muốn đến **[MASK]** để mua một **[MASK]** sữa
- Too little masking? Too expensive to train
- Too much masking? Not enough context
- Problem with MASK tokens?
  - Mask token never seen at fine-tuning

MASKING STRATEGY

- Do not always replace the masked words with **[MASK]** token
- Example: "**Chó trông rất đáng yêu**"
  - ***80%*** were replaced by the ***[MASK] token***: "Chó trông rất đáng [MASK]"
  - ***10%*** were replaced by a ***random token***: "Chó trông rất đáng chuối"
  - ***10%*** is ***unchanged***: "Chó trông rất đáng yêu"

NEXT SENTENCE PREDICTION

- Many downstream tasks are based on understanding the **relationship between two sentences**
  - Question Answer (QA) and Natural Language Inference (NLI)
- LM does not directly capture that relationship
- 50% of training data is “consecutive"



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

EVALUATION FOR BERT: GLUE

System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Average**
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

Table 1: GLUE Test results, scored by the evaluation server (`https://gluebenchmark.com/leaderboard`). The number below each task denotes the number of training examples. The "Average" column is slightly different than the official GLUE score, since we exclude the problematic WNLI set.[8] BERT and OpenAI GPT are single-model, single task. F1 scores are reported for QQP and MRPC, Spearman correlations are reported for STS-B, and accuracy scores are reported for the other tasks. We exclude entries that use BERT as one of their components.

MODEL DETAILS

Data	Wikipedia (2.5B words) + BookCorpus (800M words)	
Batch Size	131,072 words (1024 sequences * 128 length or 256 sequences * 512 length)	
Training Time	1M steps (~40 epochs)	
Optimizer	AdamW, 1e-4 learning rate, linear decay	
Architecture	Using Transformer Encoder stack	
	BERT-Base: 12-layer, 768-hidden, 12-head, 110M-params	BERT-Large: 24-layer, 1024-hidden, 16-head, 340M-params

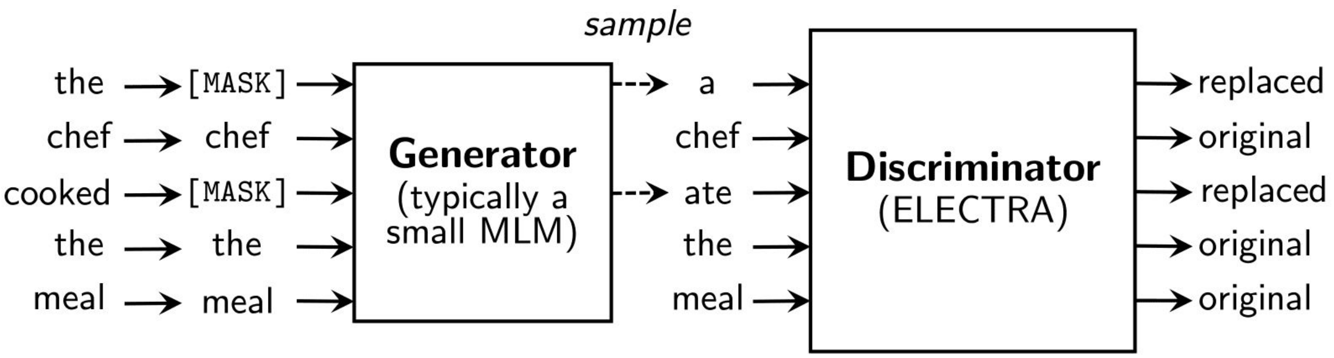
BERT VARIANTS

Comparison	BERT	RoBERTa	ELECTRA	DeBERTa
Data	16GB	160GB	16GB for small & base	160GB
Pre-trained Task	Masked Language Model Next Sentence Prediction	Masked Language Model Using Dynamic Masking	Replaced Token Detection	Replaced Token Detection w/ Efficient Embedding Sharing
Parameters	Base: 110M Large: 340M	Base: 125M Large: 355M	Small: 14M Base: 110M Large: 335M	Small: 44M Base: 98M Large: 304M
Tokenization	WordPiece	Byte-Pair Encoding (BPE)	WordPiece	Byte-Pair Encoding (BPE)
Training	Described in slide 20	Larger batchsize, longer sequences input, longer training time	Faster training	500K steps

ELECTRA

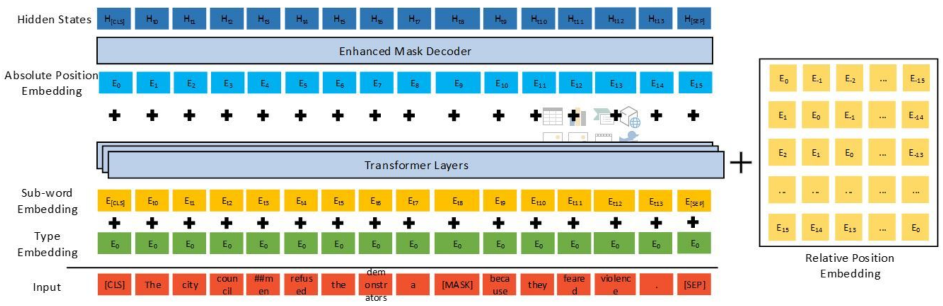
Efficiently Learning an Encoder that Classifies Token Replacements Accurately

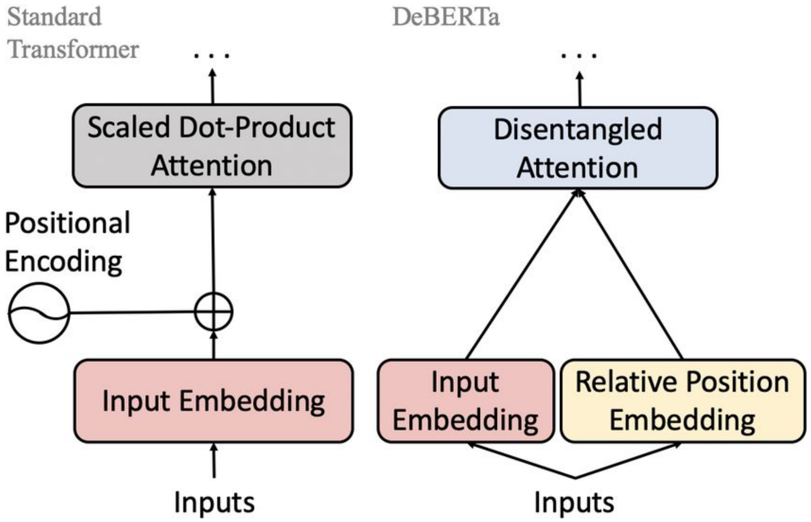
➜ Every output position is used



DeBERTaV3

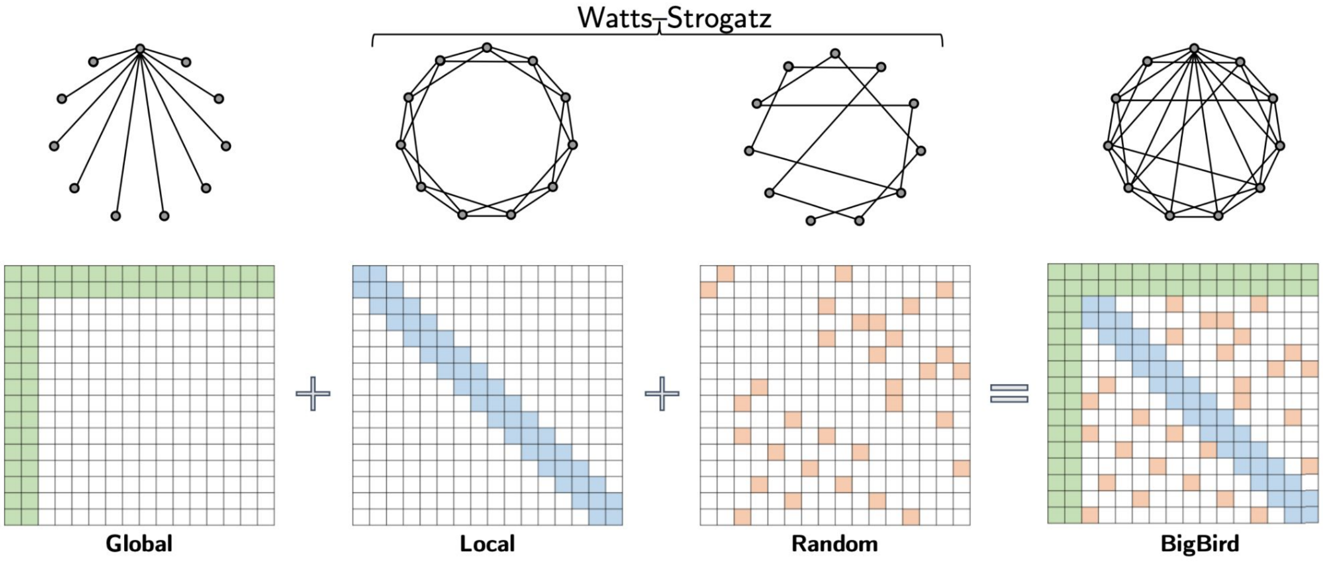
- Disentangled Attention
- Replaced Token Detection with Efficient Embedding Sharing





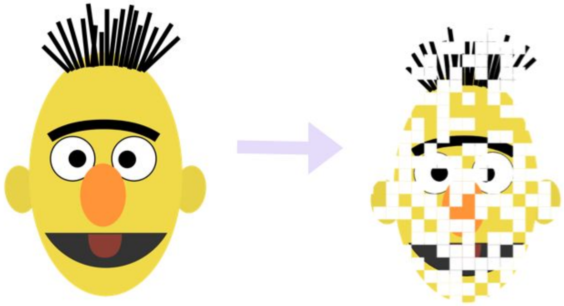
LONG TEXT

- Longformer (Beltagy et al. 2020): combine (local) sliding window and global attention
- BigBird (Zaheer et al. 2021): use sparse (global) attention mechanism
- RoFormer (Su et al. 2021): use Rotary Position Embedding



COMPACT PRE-TRAINED MODELS

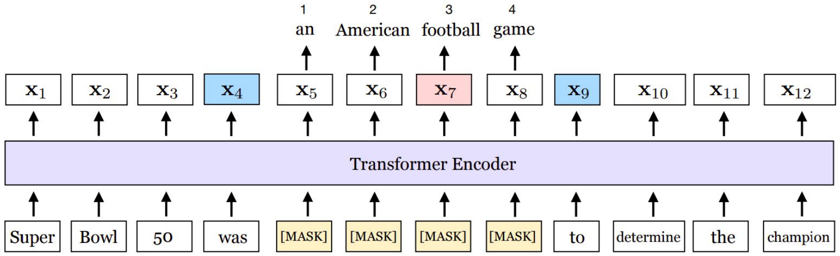
- DistilBERT: Train a model to match the distribution of BERT (Knowledge Distillation)
- TinyBERT: Same as above but uses layer-wise KD.
- MobileBERT: Same as above but uses inverted-bottleneck and blockwise KD.
- ALBERT: Smaller embeddings, and parameter sharing across all layers.

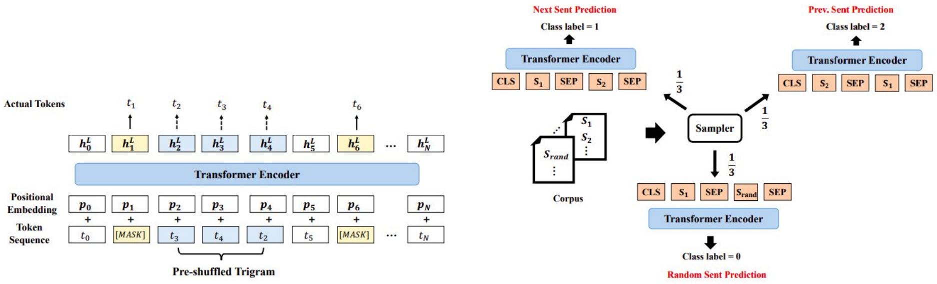


Reference: http://mitchgordon.me/machine/learning/2019/11/18/all-the-ways-to-compress-BERT.html

MORE TRAINING OBJECTIVES

- Whole Word Masking
- SpanBERT (Joshi et al., 2019) ➜ Span Masking + Span Boundary Objective
- StructBERT (Wang et al., 2019) ➜ Word-level Objective + Sentence-level Objective





(a) Word Structural Objective
(b) Sentence Structural Objective

Vietnamese BERT

Comparison	PhoBERT	viELECTRA	ViDeBERTa
Data	20GB (Wiki+News)	60GB (Oscar+News)	138GB CC100
Pre-trained Task	Masked Language Model	Replaced Token Detection	Replaced Token Detection w/ Efficient Embedding Sharing
Parameters	Base Large	Base: 125M	Xsmall: 22M Base: 86M Large: 304M
Tokenization	Word-level fastBPE	Word Piece*	Word-level BPE
Training	256len/500K steps	100 epochs	500K steps

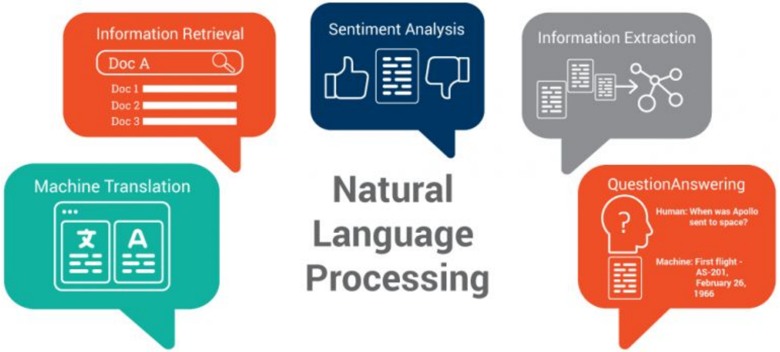
Downstream Tasks

EVALUATION FOR BERT: GLUE

- General Language Understanding Evaluation (**GLUE**) benchmark: Standard split of data to train, validation, test, where labels for the test set is only held in the server.
  - https://gluebenchmark.com/
- Sentence pair classification
  - **MNLI**, Multi-Genre Natural Language Inference
  - **QQP**, Quora Question Pairs
  - **QNLI**, Question Natural Language Inference
  - **STS-B**, The Semantic Textual Similarity Benchmark
  - **MRPC**, Microsoft Research Paraphrase Corpus
  - **RTE**, Recognizing Textual Entailment
  - **WNLI**, Winograd NLI is a small natural language inference dataset
- Single sentence classification
  - **SST-2**, The Stanford Sentiment Treebank
  - **CoLA**, The Corpus of Linguistic Acceptability

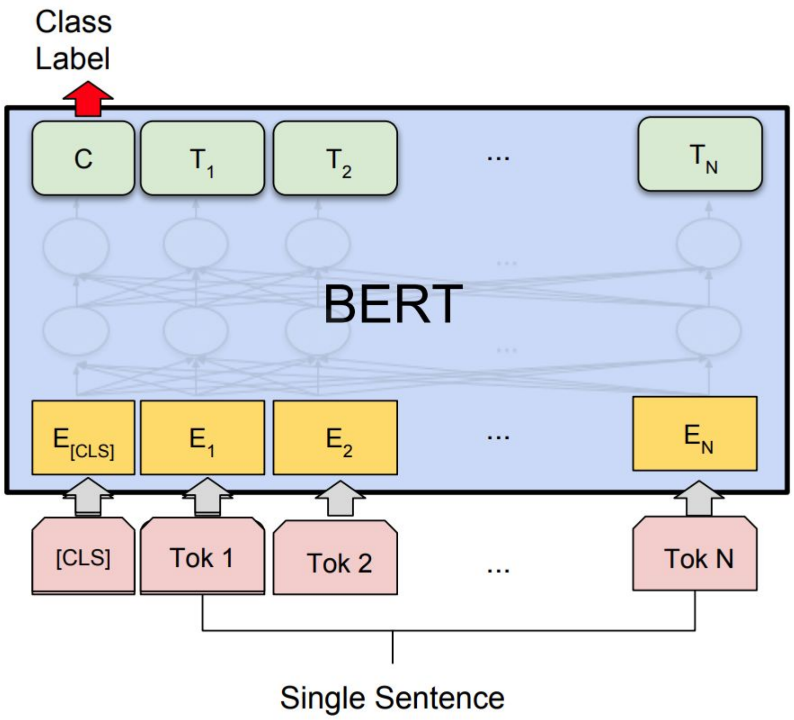
COMMON DOWNSTREAM TASKS

- Sentiment Analysis
- Named Entity Recognition
- Natural Language Inference
- Question Answering



SINGLE SENTENCE CLASSIFICATION

- **Sentiment Analysis**
  - Tôi thích đọc sách -> Positive
  - Tôi ghét phim kinh dị -> Negative
- **Input**: [CLS] Tôi thích đọc sách [SEP] -> Positive/Negative?



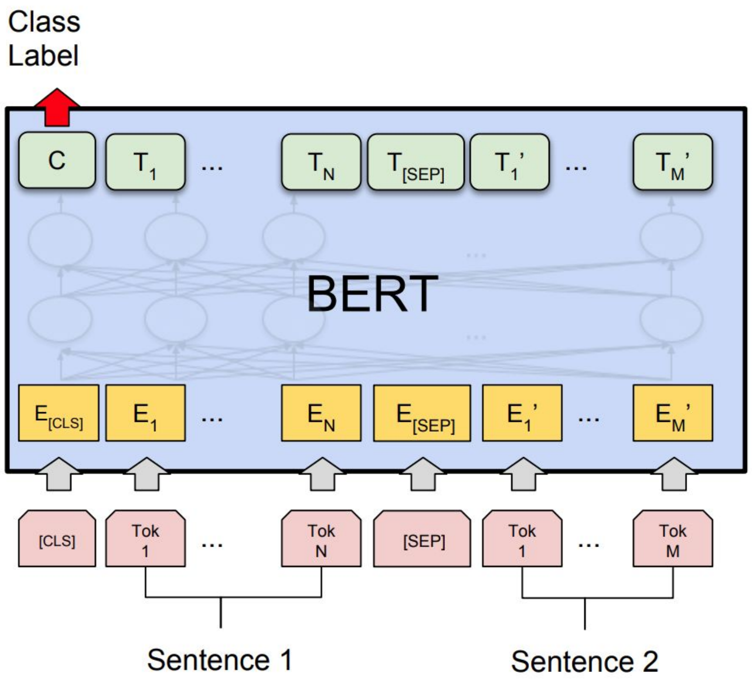
Other applications: Aspect-based Sentiment Analysis, Spam Detection, etc.

Source: https://arxiv.org/pdf/1810.04805.pdf

SENTENCE-PAIR CLASSIFICATION

- **Natural Language Inference**
  - Sent 1: Tôi làm nhiều việc tốt
  - Sent 2: Tôi được mọi người yêu quý
  - Sent 1 -> Sent 2: Entailment
- **Input**: [CLS] Tôi làm nhiều việc tốt [SEP] Tôi được mọi người yêu quý [SEP] -> Entailment/Contradiction/Neutral?

Other applications: Semantic Similarity, Ranking, etc.

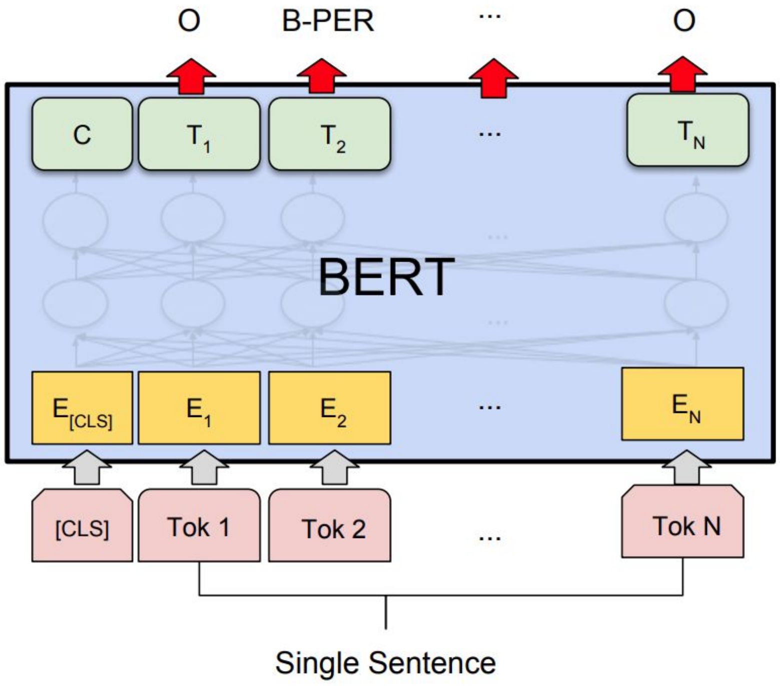


Source: https://arxiv.org/pdf/1810.04805.pdf

TOKEN CLASSIFICATION

- **Named Entity Recognition (NER)**: detect entities such as PER, LOC, ORG
- **Text**: Mark Zuckerberg cho ra đời Facebook trong phòng ký túc xá của mình tại Đại học Harvard vào ngày 4 tháng 2 năm 2004.
- **Input**: [CLS] Text [SEP]
- **Output**: B-PER, I-PER, O, O,...

Other applications: POS Tagging, Punctuation Prediction, etc.

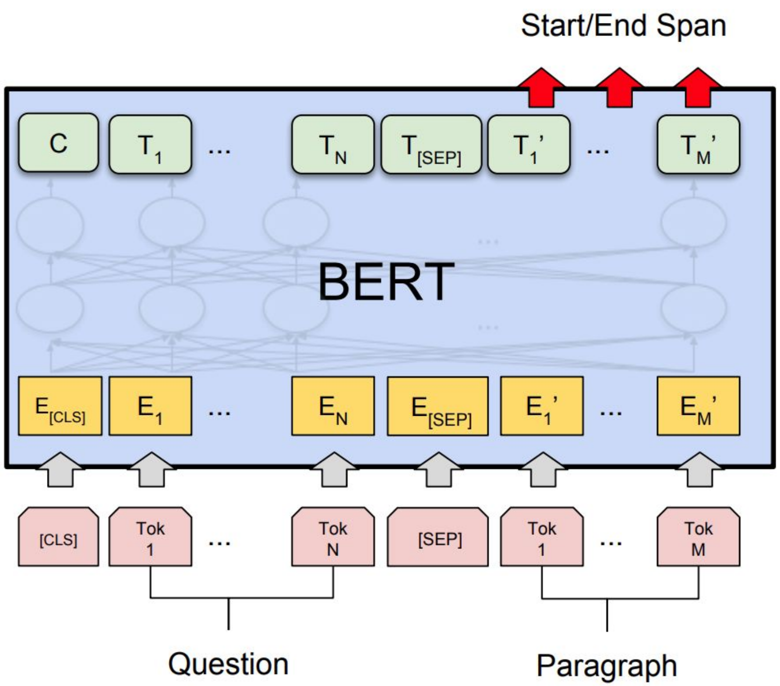


Source: https://arxiv.org/pdf/1810.04805.pdf

QUESTION ANSWERING

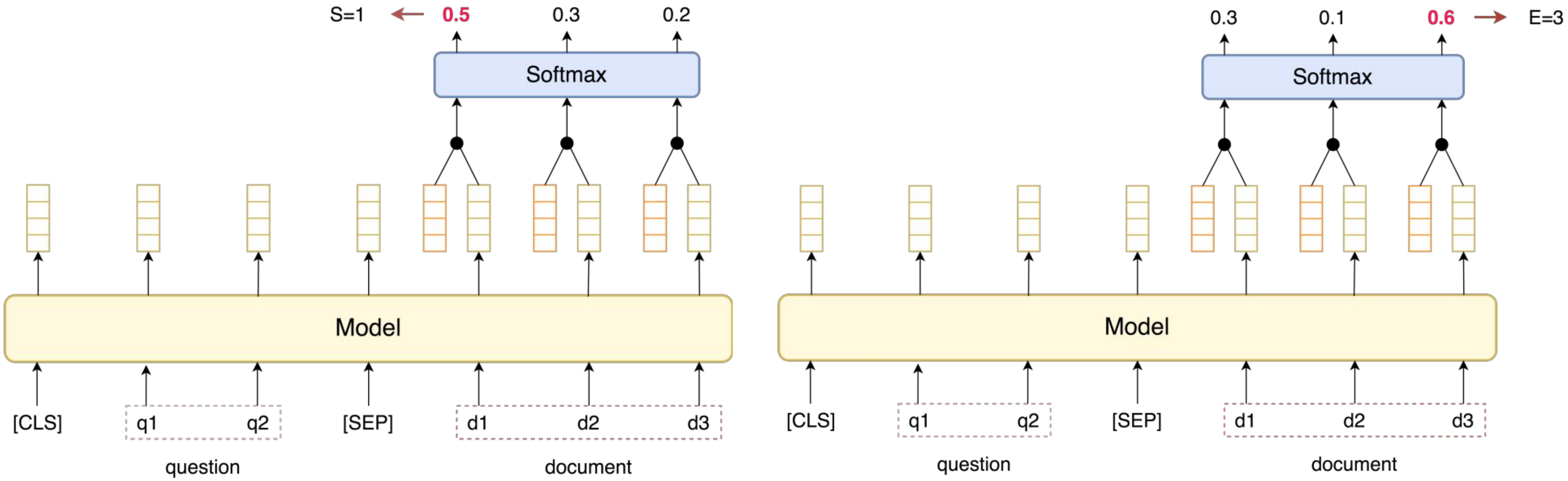
VietAI

- **Detect answer span** in paragraph
- **Question**: Đâu là thủ đô của Việt Nam?
- **Paragraph**: Hà Nội là thủ đô của nước Cộng hoà Xã hội chủ nghĩa Việt Nam, cũng là kinh đô của...
- **Input**: [CLS] Question [SEP] Paragraph [SEP]
- **Output**: start=0, end=2
- **Answer**: Hà Nội



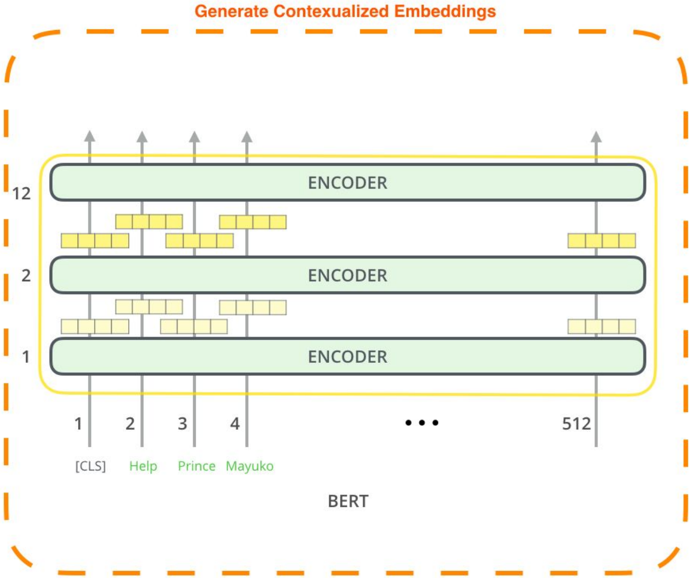
Source: https://arxiv.org/pdf/1810.04805.pdf

QUESTION ANSWERING

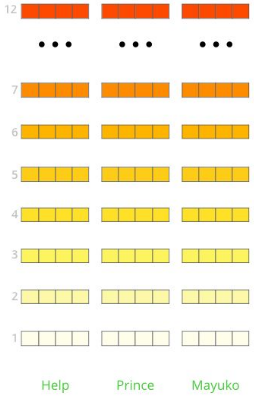


FEATURE EXTRACTION

- Which layer should be used?



The output of each encoder layer along each token's path can be used as a feature representing that token.

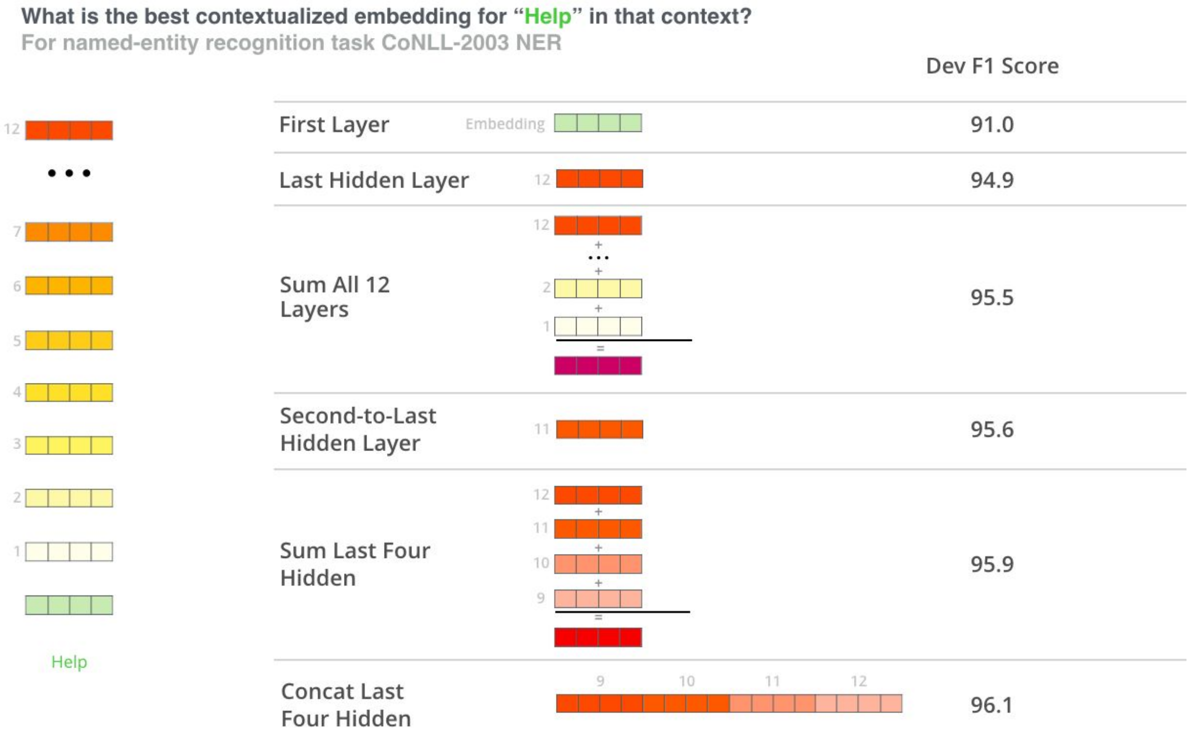


But which one should we use?

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

FEATURE EXTRACTION

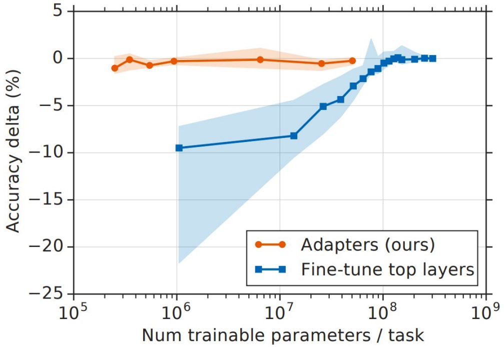
- Depend on the task

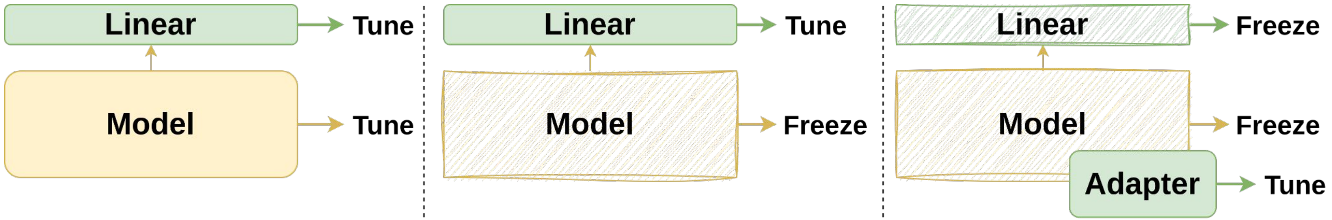


Source: https://jalammar.github.io/illustrated-bert/

FINE-TUNING STRATEGY

- Train the entire architecture
- Train some layers while freezing others
- Freeze all layers
- Freeze all - train an adapter



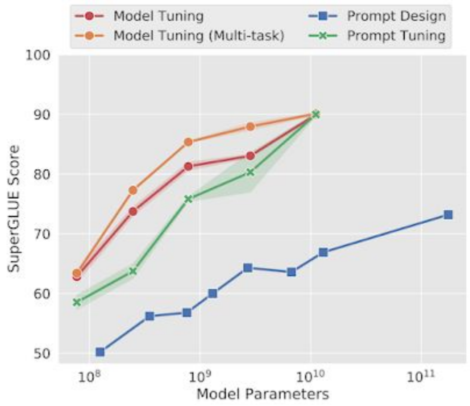


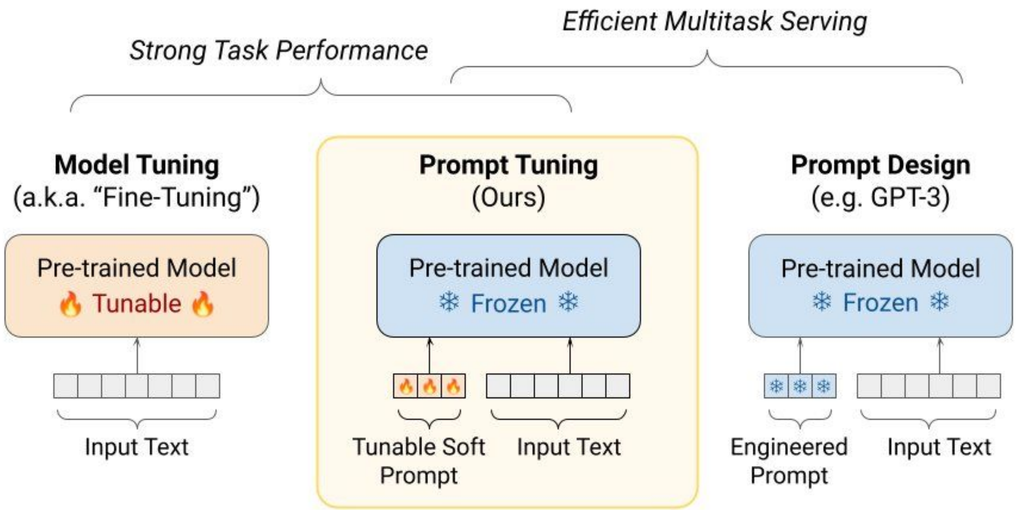
Why doesn't freeze all BERT layers during fine-tuning tasks?

Reference: https://arxiv.org/abs/1902.00751

PROMPT TUNING

Instead of fine-tuning, hold the model fixed and tune a fixed-size vector representation.





Reference: https://ai.googleblog.com/2022/02/guiding-frozen-language-models-with.html

WHAT WE LEARN TODAY

- The different between word embeddings: Word2Vec, GloVe, CoVe, ELMo
- Types of learning: multi-task, transfer learning, pretraining.
- The architecture of BERT and its variants
- Introduction of downstream tasks: NLI, QA, NER, etc

Thank you!