COMP4650/6490 Document Analysis

Pre-trained Language Models — Part I

ANU School of Computing

Word Representations

We have looked at word representation

- Learnt from context words, e.g. word2vec

This has limitations, because:

- The representation doesn't depend on the context in which the word instance occurs
- The same word can have different meanings, syntactic behaviour and connotations which cannot be captured by a single representation

Recap

We have used DNNs to capture context and structure in sentences

- Sequential structure
- Unknown structure

Training a DNN is expensive, in terms of:

- Computation
- Data

Contents

- Self-supervised learning: training with unlabelled / self-labelled data
- Pre-trained language models
- Transfer learning through fine-tuning

Contents

- Self-supervised learning: training with unlabelled / self-labelled data
- Pre-trained language models
- Transfer learning through fine-tuning

Self-supervised Learning

What is self-supervised learning?

- Use naturally existed supervision signals for training Examples:
  - Predict the next word in the text.
  - Predict the colour of a nearby pixel in an image.
- (Almost) no human effort to construct training data

Why do we need it?

- To train large models without spending millions of dollars on creating training data
- To pre-train models for later fine-tuning on a data-poor task

Self-supervised Learning

Like *unsupervised* learning we don't explicitly need to label our data.

Like *supervised* learning we have labels (implicit in the data)

Self-supervised learning is more like supervised learning than unsupervised learning

How can we do self-supervised learning?

Self-supervised Learning

How can we do self-supervised learning?

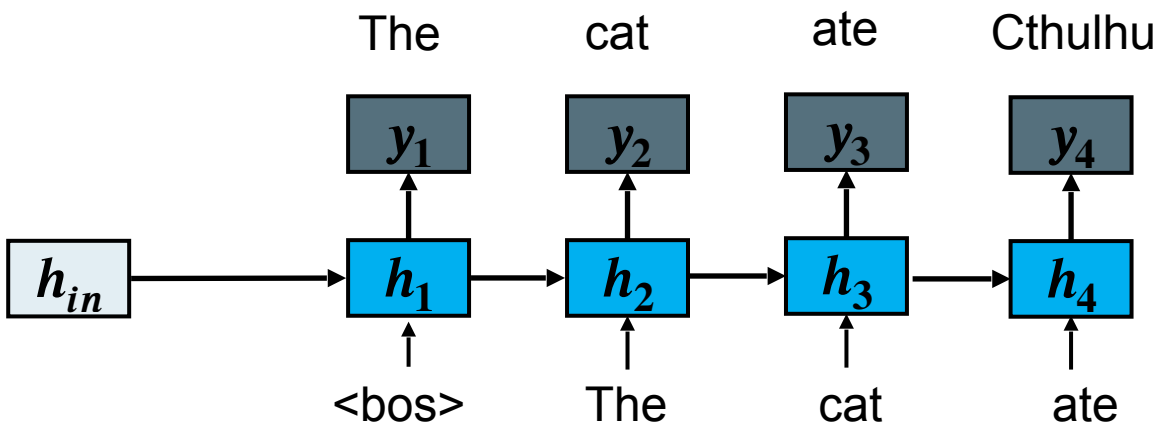
word2vec does self-supervised learning

- It tries to predict context words given a central word
- It doesn't require labelled data
- We can use any text we have lying around

Neural Language Model

RNN language models are also self-supervised

- Predict the next word in the sentence given all previous words
- Train using sentences from anywhere (e.g. Wikipedia, Reddit, Canberra times)

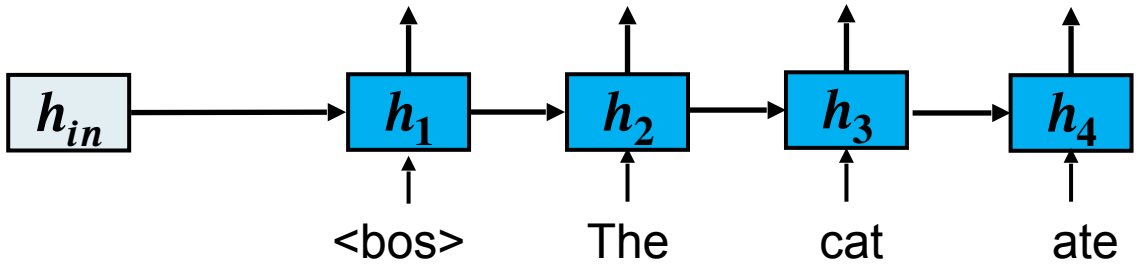


At training time we feed the ground truth word in at the next step.

Context Specific Word Representations

The RNN language models are producing context-specific word representations at each position.

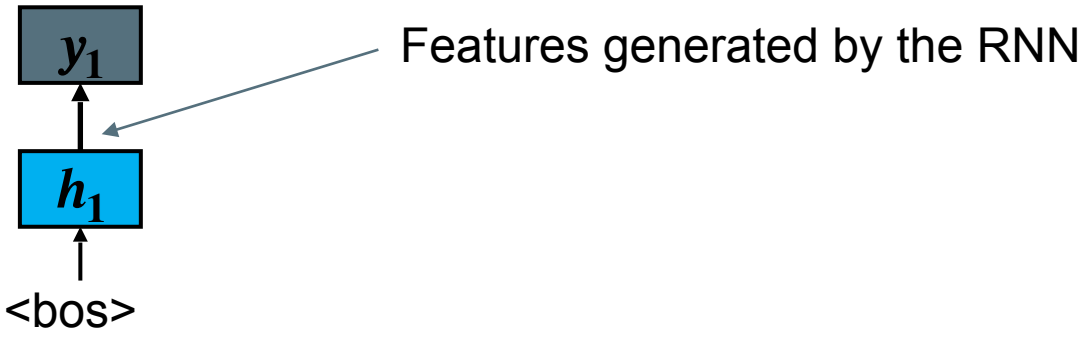
We can use these representations for other tasks.



Context Specific Word Representations

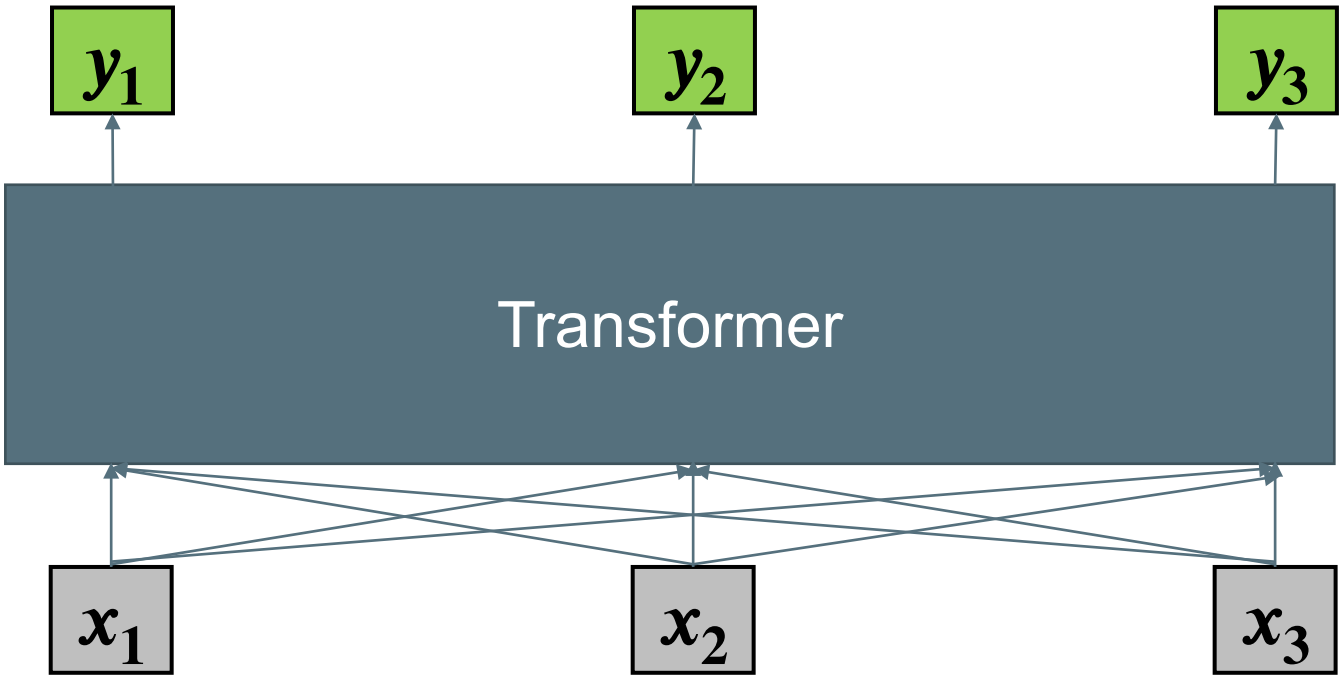
Feature Learning:

- Earlier in the course we mentioned neural networks can learn feature representations with each layer.
- We can use these learnt features by removing the last layer of the network. (we can also get features from other layers if available)



Context Specific Word Representations

Transformers for language also produce context specific word representations which we can use for other tasks.



Contents

- Self-supervised learning: training with unlabelled / self-labelled data
- Pre-trained language models
- Transfer learning through fine-tuning

Transfer Learning

We have an NLP task in mind (e.g. sentiment classification, NER). Normally we would train a model (e.g. an RNN) on a (usually small) labelled dataset.

Transfer learning approach:

1. Add task-specific layers on top of a pre-trained model
2. Fine-tuning using a small amount of labelled data for the specific NLP task
   - To learn parameters of the task-specific layers
   - Freeze or make minimal adjustments to parameters of the pre-trained model (e.g. using a small learning rate)

ELMo: Embeddings from Language Models

Peters et al. Deep contextualized word representations. 2018.

- Learns a deep Bi-RNN and uses all its layers to give *contextual word vectors.*
- No fixed context window. Bounded by memory capacity of the network.

Applying to new problems:

- Find a state-of-the-art model for your problem.
- Input ELMo embeddings rather than other word embeddings.

ELMo

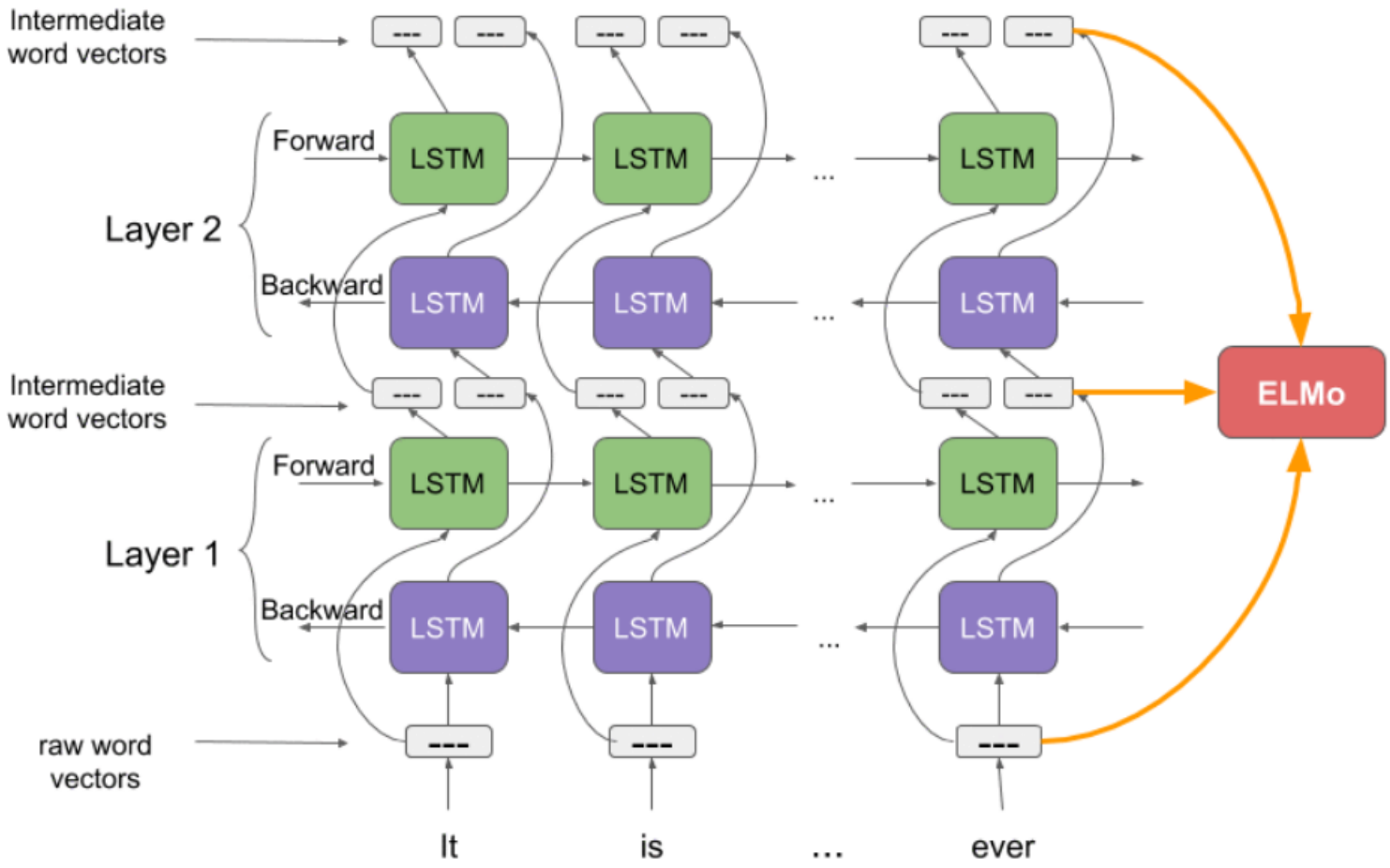


Image src https://www.analyticsvidhya.com/blog/2019/03/learn-to-use-elmo-to-extract-features-from-text/

ELMo: Results

Including contextual embeddings from ELMo in these existing state-of-the-art models significantly improves performance.

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%

Transformer Models

These models are based on Transformers:

GPT	**BERT**	**GPT-2**	**GPT-3**	**GPT-4**
June 2018	Oct 2018	Feb 2019	June 2020	March 2023
Training 240 GPU days	Training 256 TPU days ~320-560 GPU days	Training ~2048 TPU v3	Training ~$4.6M worth of Tesla V100 cloud instance	Training > $100M ?
117 Million parameters	168 Million parameters	1.5 Billion parameters	175 Billion parameters	> 1 Trillion?
by OpenAI	by Google AI	by OpenAI	by OpenAI	by OpenAI

GPT Models

- GPT models are language models.
  - GPT-1,2,3 are trained to predict the next token in a sequence
  - Similar architectures:
    Masked multi-head self-attention (decoder-only) transformers (with 12, 48, 96 blocks, respectively)
  - GPT-4: Multimodal (input: text and/or image, output: text)
- GPT models did several things differently to previous models.
  - Used Transformers
  - Used much more data
  - Used much more compute
  - A few other tricks.

GPT Models: Dataset

GPT

BookCorpus dataset (free books from smashwords.com) ~1GB text

GPT-2

WebText dataset: Every web-page that has been linked to from Reddit with a rating of at least +3

- 3 people have deemed this link to be interesting/funny/informative.
- Contains 40GB of raw text.

GPT-3

Common Crawl (filtered), WebText2, Books1/2, Wikipedia (en)

GPT-4

Licensed data

GPT Models: Pre-training vs Fine-tuning

Pre-training:

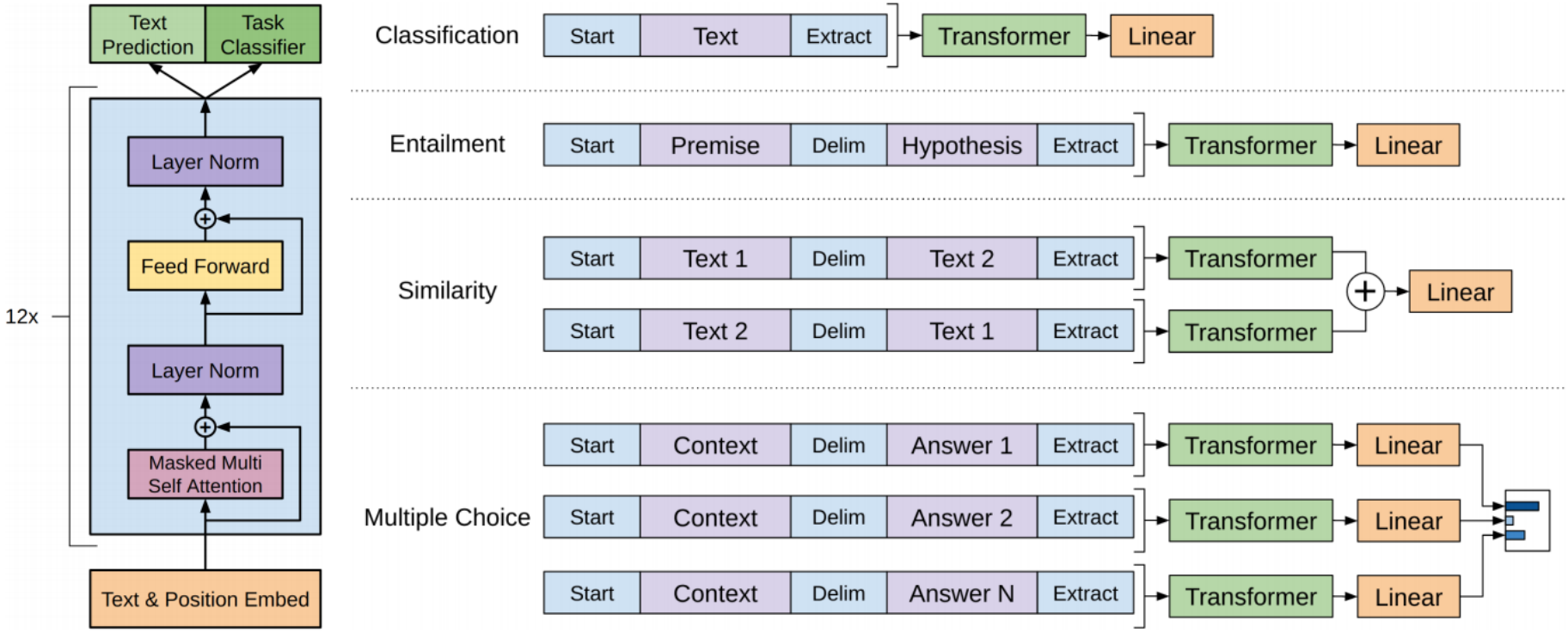
- Next word prediction.
- Cross-entropy loss.
- Large amount of data (self-supervised).
- Only needs to be done once.

Fine-tuning:

- Start the parameters from where pre-training finished.
- Normally uses a small learning rate
- Task specific loss
- Task specific data
- Needs to be done for every task
- May add extra layers compared to pre-training

GPT Models: Fine-tuning on New Tasks

- Modify the input structure depending on the task
- Fine-tune the model on task specific data



Radford et al. Improving language understanding by generative pre-training. 2018.

BERT

BERT stands for ***B**idirectional **E**ncoder **R**epresentations from **T**ransformers*

Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. 2018.

Dataset: BookCorpus and Wikipedia (en) ~16GB text
(RoBERTa: better training with ~160GB text)

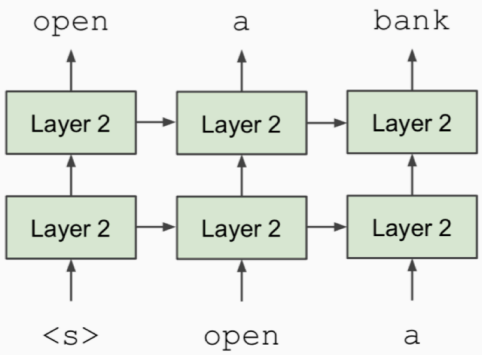
Learning: Optimise a combined loss

- Masked Language Modelling
- Next Sentence Prediction

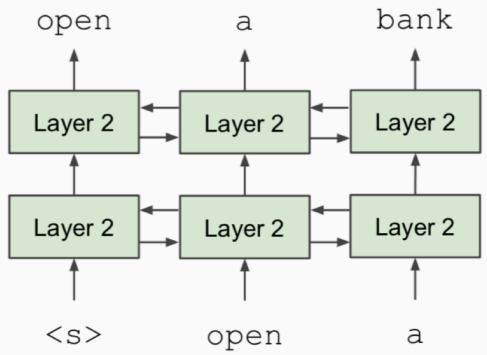
Bidirectional vs Unidirectional

- **Problem**: Language models only use left context *or* right context, but language understanding is *bidirectional*.

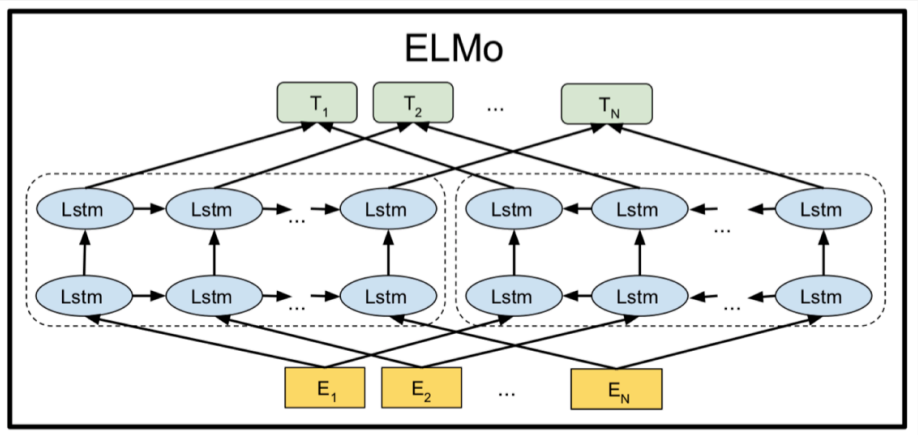
Unidirectional context
Build representation incrementally

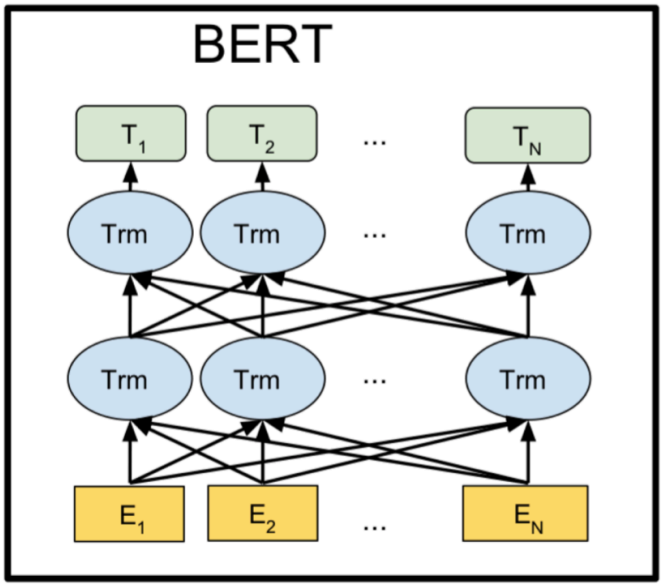


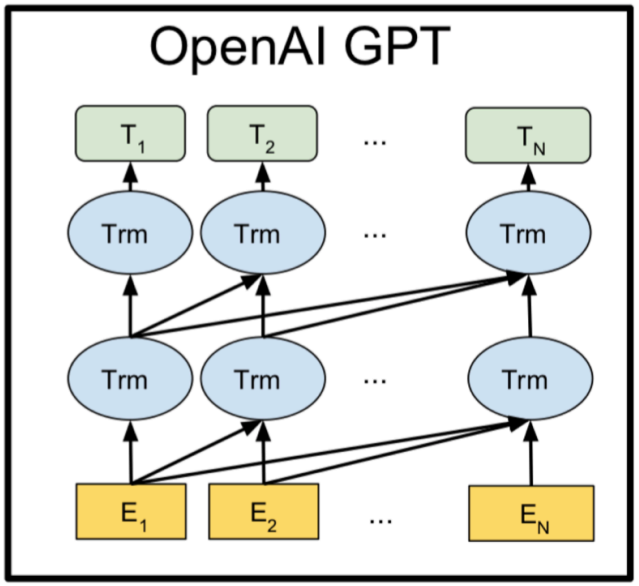
Bidirectional context
Words can "see themselves"



Architecture Comparison







BERT Learning: Mask out

- What is the probability of Pr(word | context)?
- Mask out *k*% of the input words, and then predict the missing words (*k*=15 in practice)

store bottle

the man went to the [MASK] to buy a [MASK] of milk

  - 80% of masked words are replaced with [MASK]
  - 10% are replaced with another random word
  - 10% are left the same

BERT Learning: Next Sentence Prediction

- To learn *relationships* between sentences, predict whether *Sentence B* is an actual sentence that proceeds *Sentence A*, or a random sentence

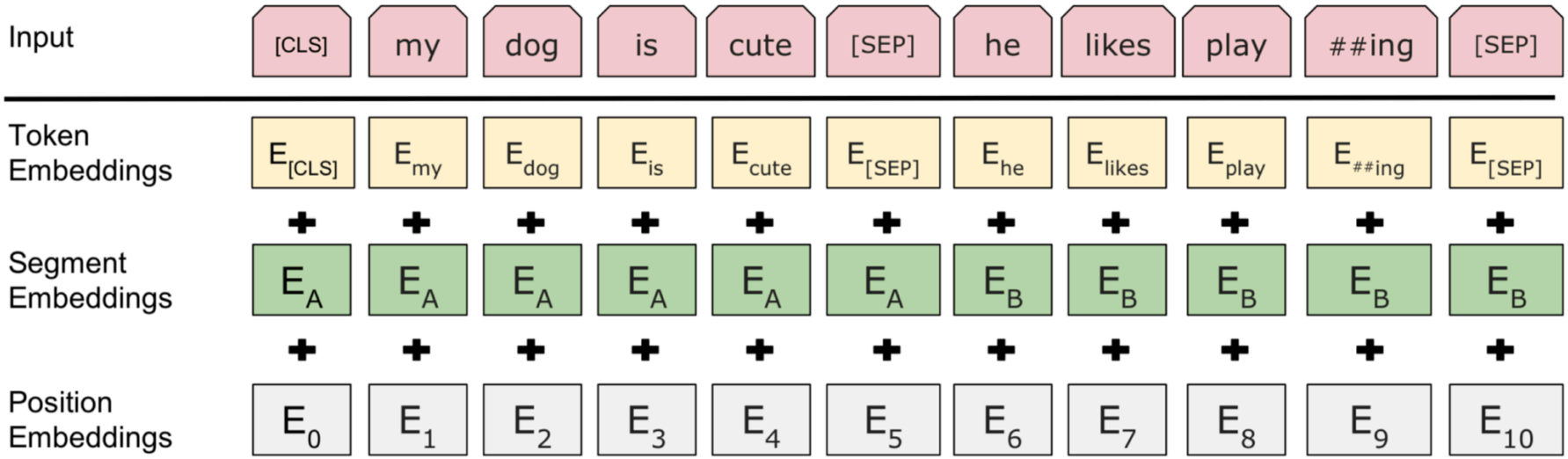
Sentence A = `The man went to the store.`
Sentence B = `He bought a gallon of milk.`
Label = `IsNextSentence`

Sentence A = `The man went to the store.`
Sentence B = `Penguins are flightless.`
Label = `NotNextSentence`

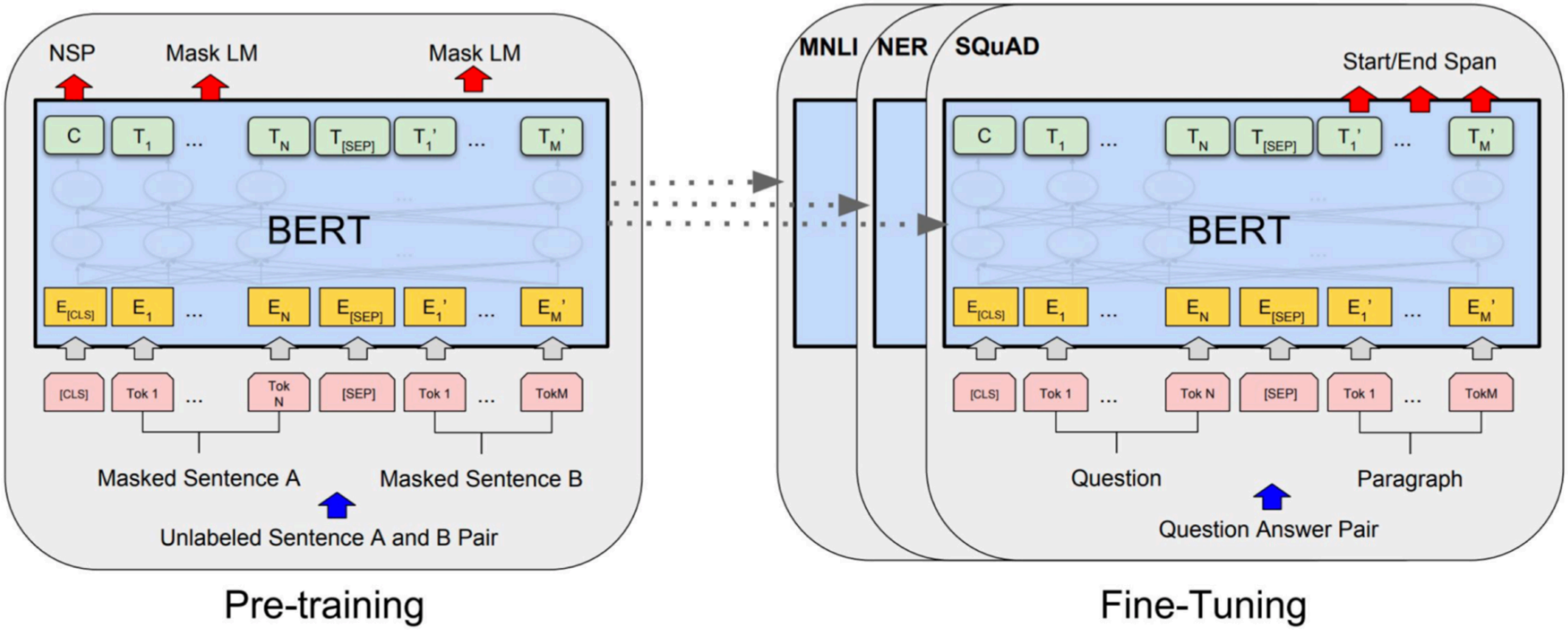
BERT Learning: Next Sentence Prediction

Sentence pair encoding:

- Token embeddings are for word pieces
- Learned segmented embedding represents each sentence
- Positional embedding is as for other Transformer architectures

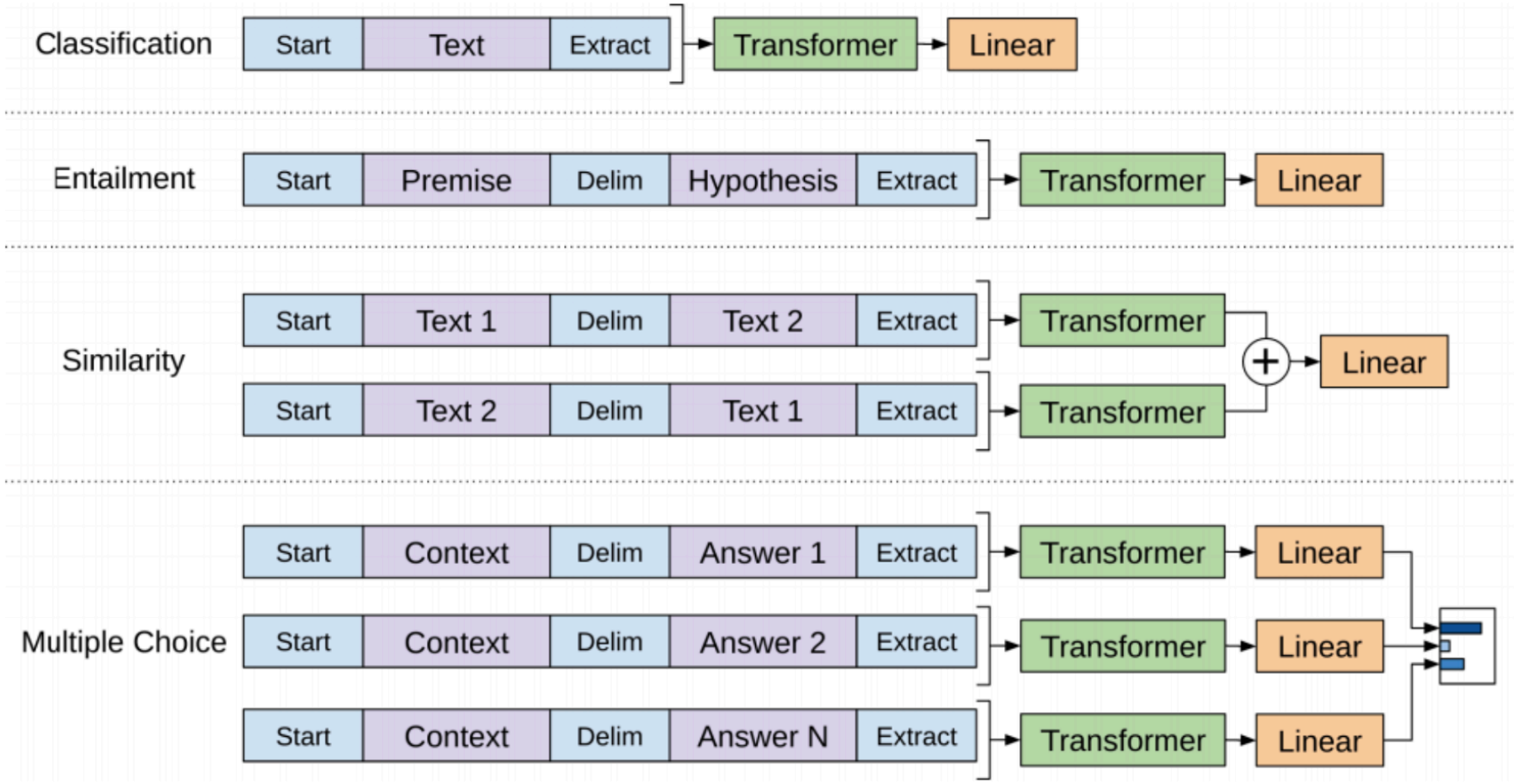


BERT: Pre-training vs Fine-tuning



Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. 2018

Using BERT on Multiple Tasks



BERT: Results on GLUE Tasks

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	**Average** -
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	79.9	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	88.1	91.3	45.4	80.0	82.3	56.0	75.2
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.1	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**91.1**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**81.9**

XLNet

- A regular language model only predicts the next token in a sequence for one ordering.
- XLNet randomly samples an ordering then constructs the sequence in that order.

XLNet

For example,

- Given sentence “quick brown fox”
- Sample random order 2,1,3

(1) Given no information, try to predict the word at position 2.

(2) Given that the word at position 2 is “brown”, try to predict the word at position 1.

(3) Given that the word at position 2 is “brown” and the word at position 1 is “quick”, try to predict the word at position 3.

Summary

- We can train large language models using text on the web
- We can use the outputs of these models as drop-in replacements for other word-vectors
- Or we can make small modifications then fine-tune these models for specific tasks

References

- Chapter 11, Speech and Language Processing
- ELMo - https://arxiv.org/abs/1802.05365
- GPT - https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf
- GPT-2 - https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf
- GPT-3 - https://arxiv.org/abs/2005.14165
- GPT-4 - https://arxiv.org/abs/2303.08774
- BERT - https://arxiv.org/abs/1810.04805
- RoBERTa - https://arxiv.org/abs/1907.11692
- XLNet - https://arxiv.org/abs/1906.08237