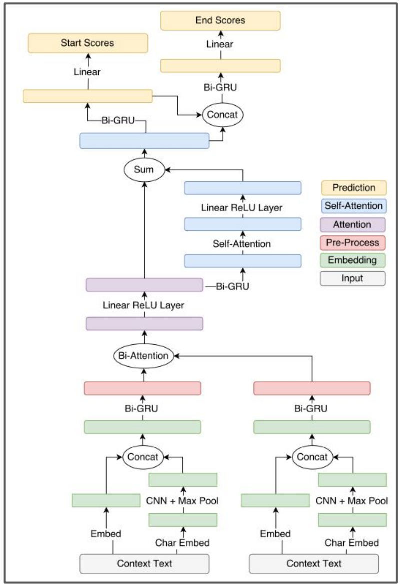
Improving Language Understanding by Generative Pre-Training

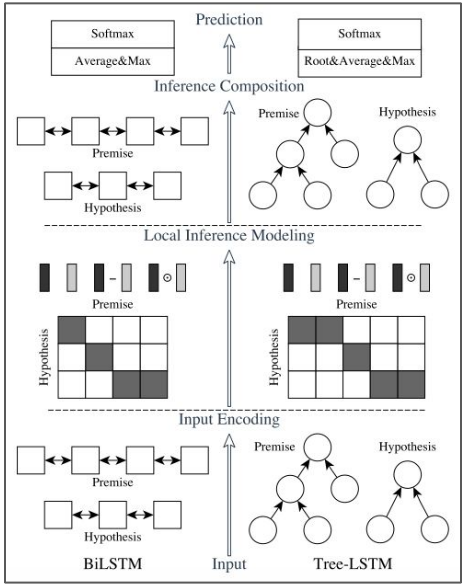
Motivation

- Semi-supervised learning: embeddings
    - Unsupervised learning of word-level or phrase-level stats
        - E.g. Word embeddings, ELMo vectors
    - Supervised training using these word-level features
        - ELMo Example:
            - Question Answering: Add ELMo to modified BiDAF model
            - Textual Entailment: Add ELMo to ESIM sequence model
            - Coreference Resolution: Add ELMo to end-to-end span-based neural model

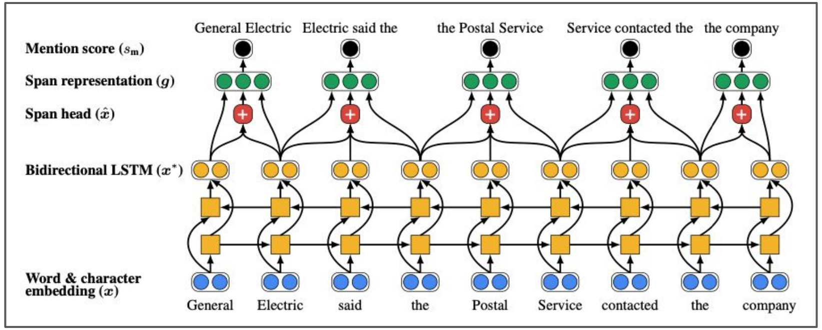
ELMo: Different Models for Each Task



Question Answering



Textual Entailment

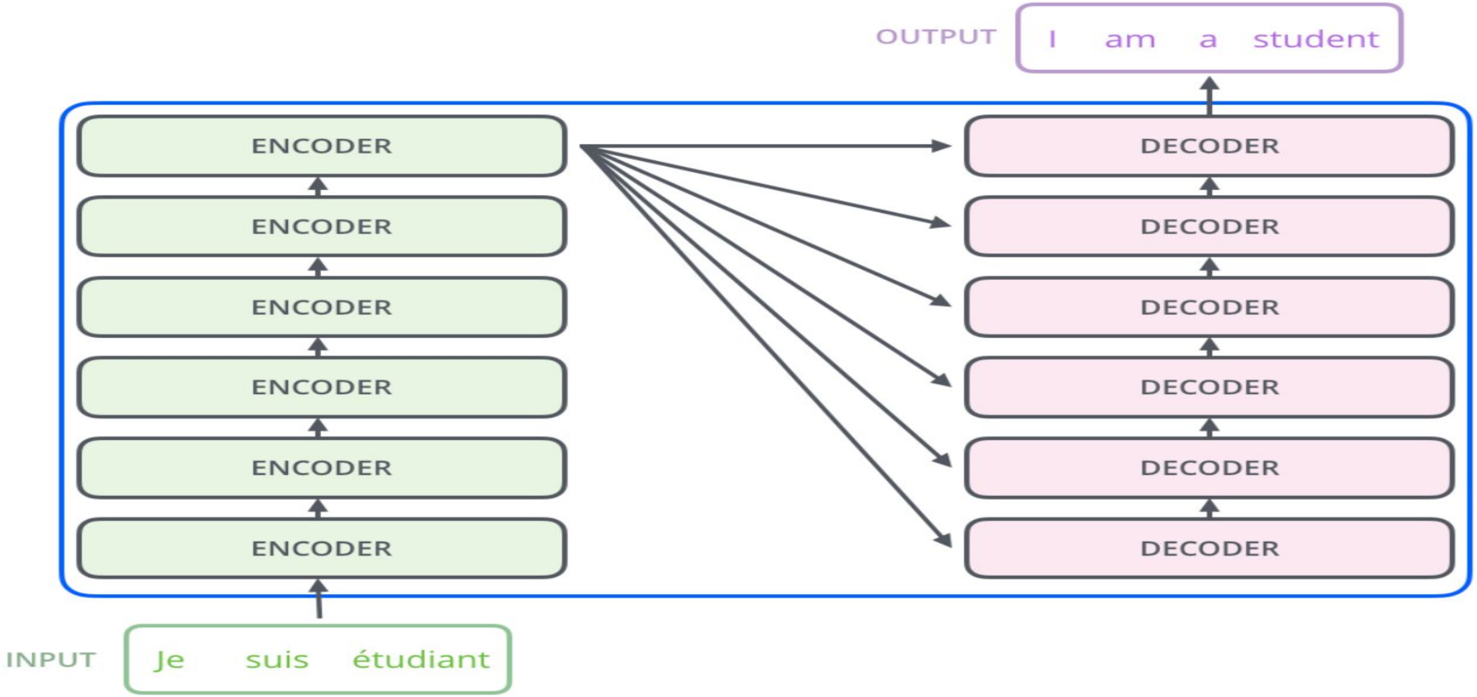


Coreference Resolution

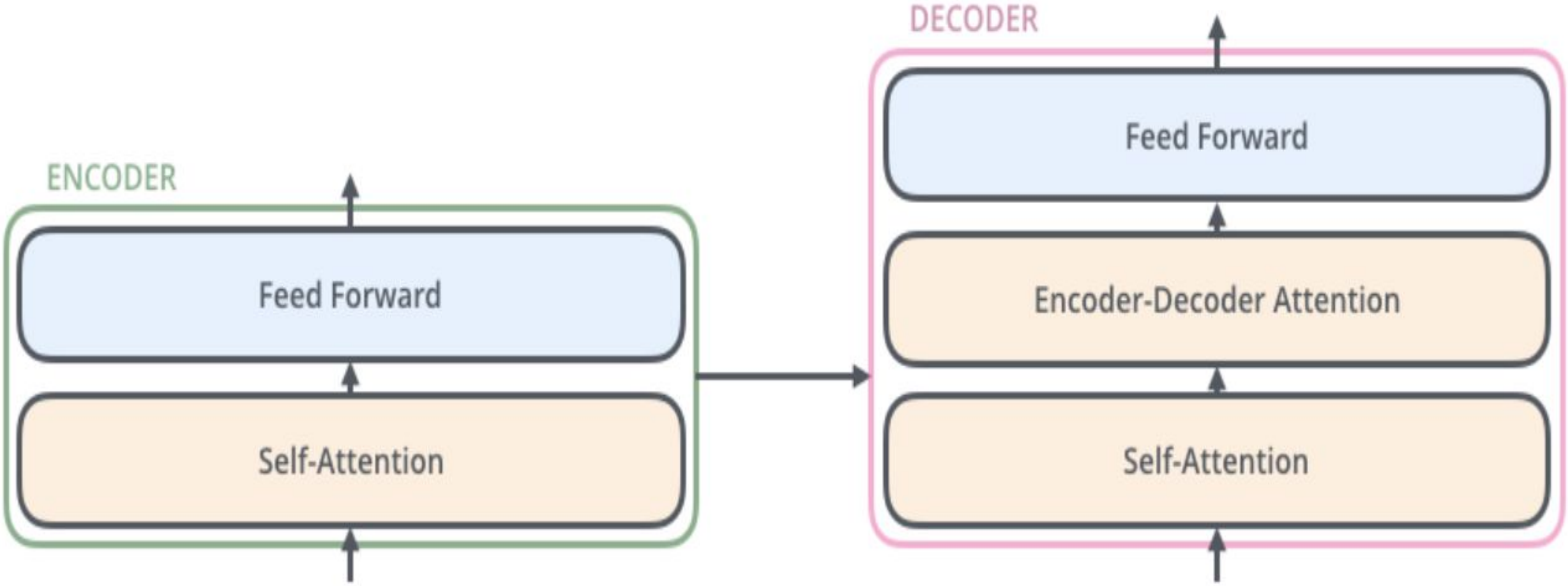
Generative Pre-Training (GPT)

- Single Model: Transformers
  - Make longer-distance connections
  - Faster training
- Unsupervised pre-training
  - Similar objective as Word2Vec
  - Predict context words
- Supervised fine-tuning
  - Use pre-trained model
  - *Only swap the last layer*
- Takeaways
  - Apply one pre-trained model to many tasks
  - BPE Tokens
  - Pre-trained Transformers learn something, even with no supervision

Transformer

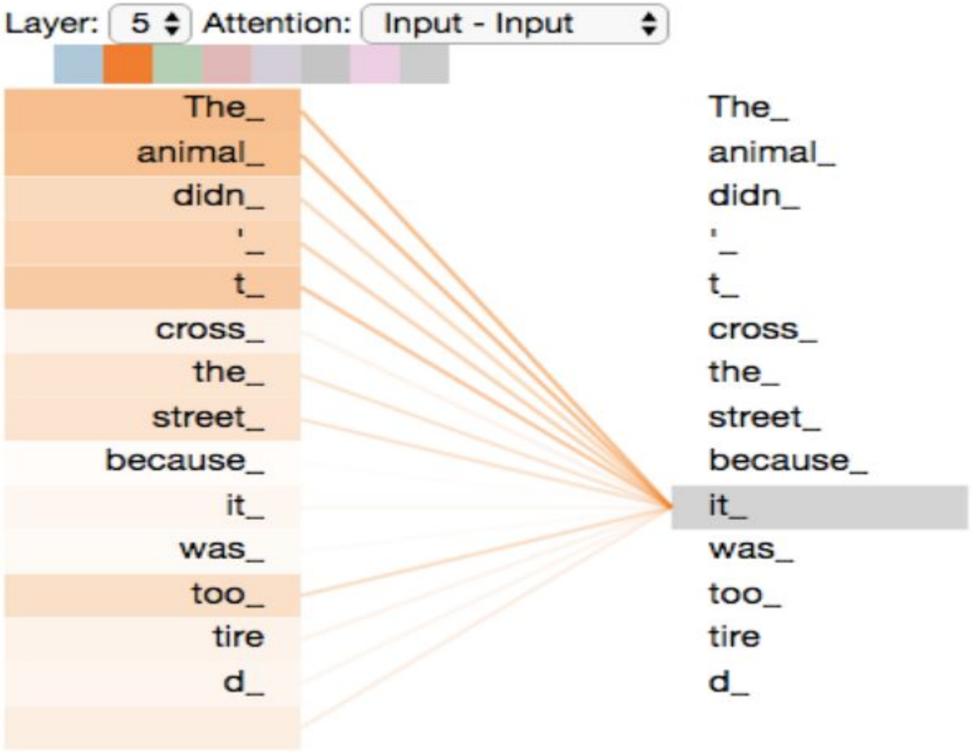


Transformer Encoder and Decoder

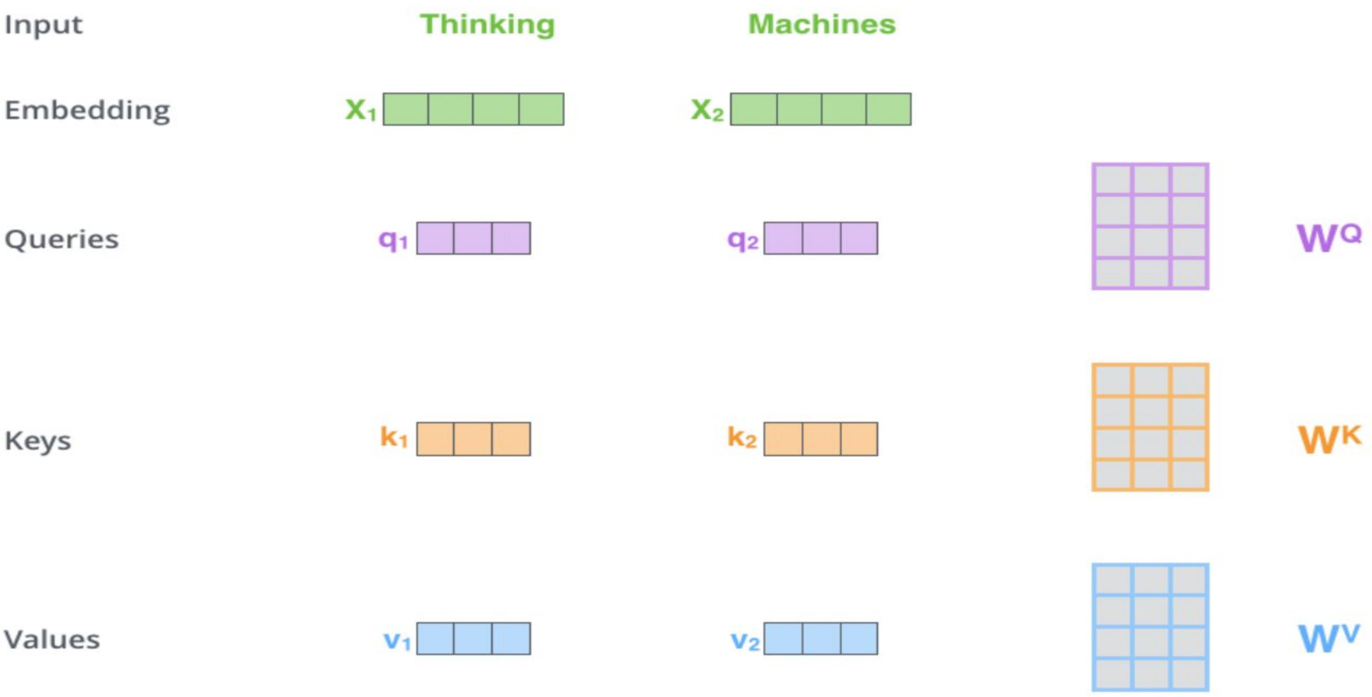


Transformer is more efficient than LSTM because it lends itself to parallelization.

Self-Attention

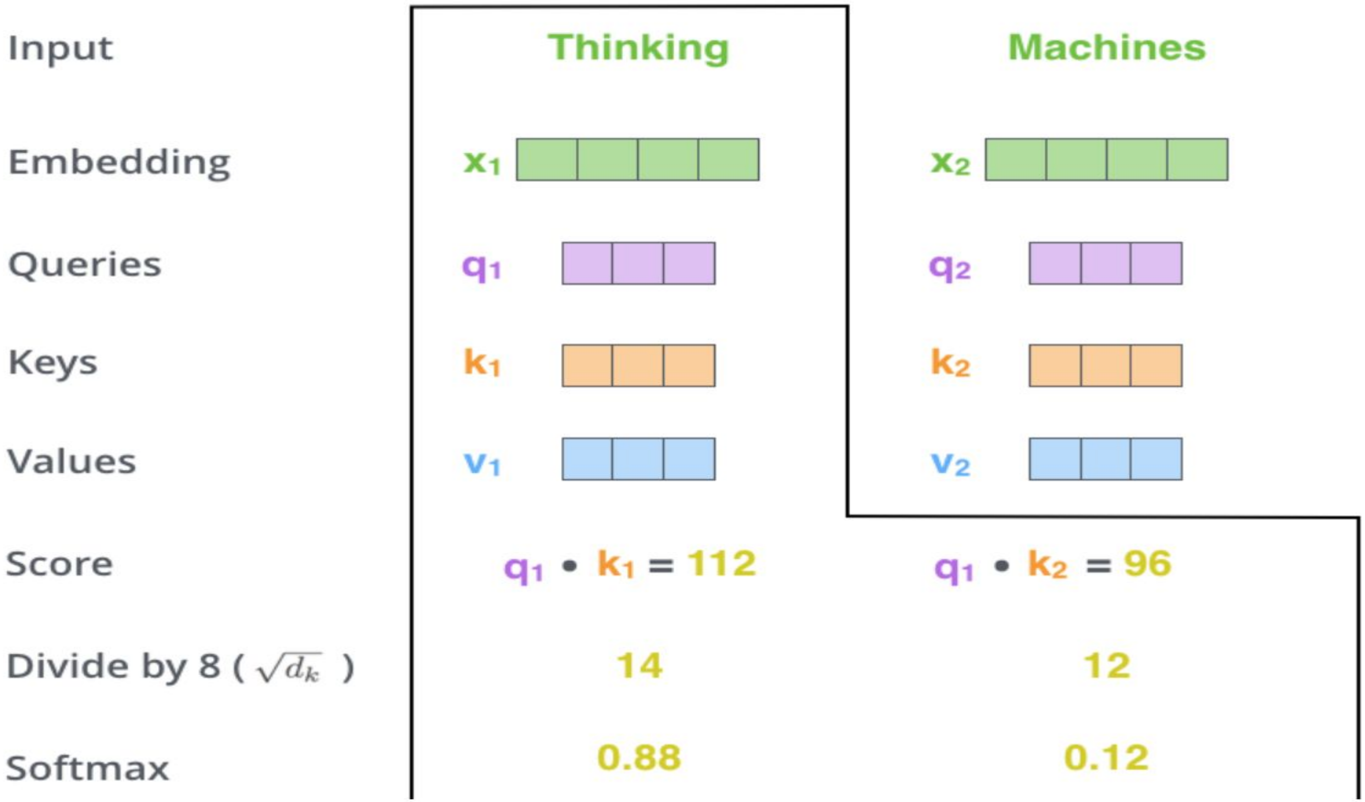


Self-Attention in Detail



Multiplying x1 by the WQ weight matrix produces q1, the "query" vector associated with that word. We end up creating a "query", a "key", and a "value" projection of each word in the input sentence.

Self-Attention in Detail



Transformers

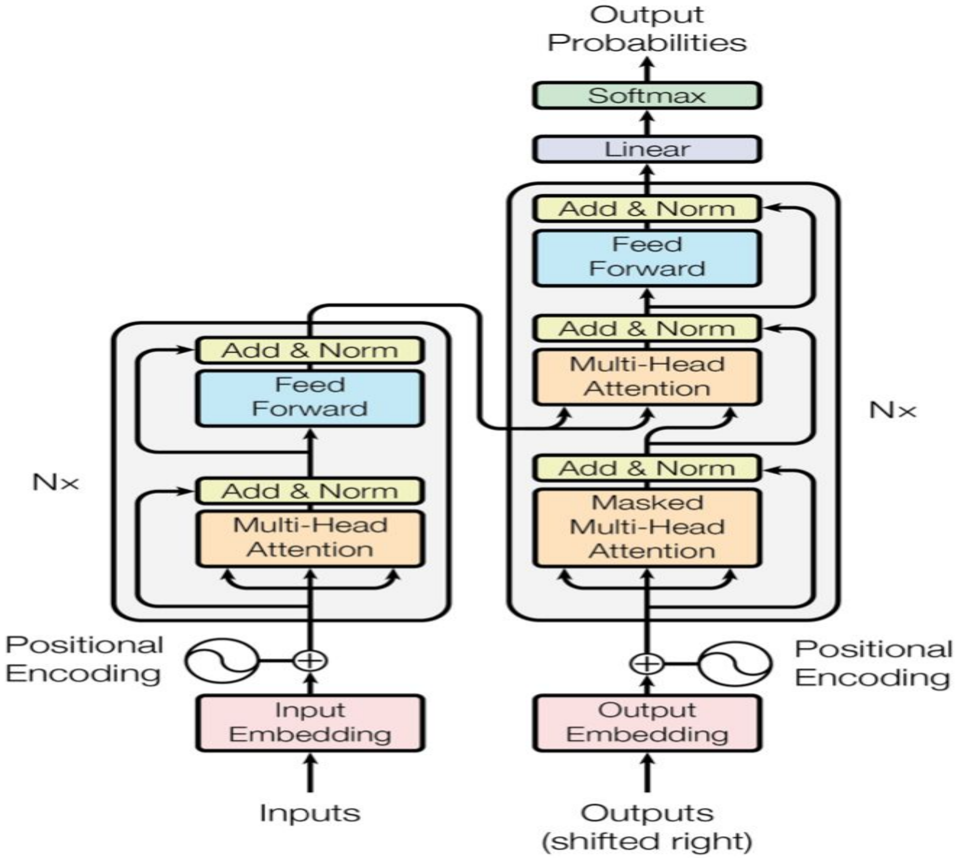
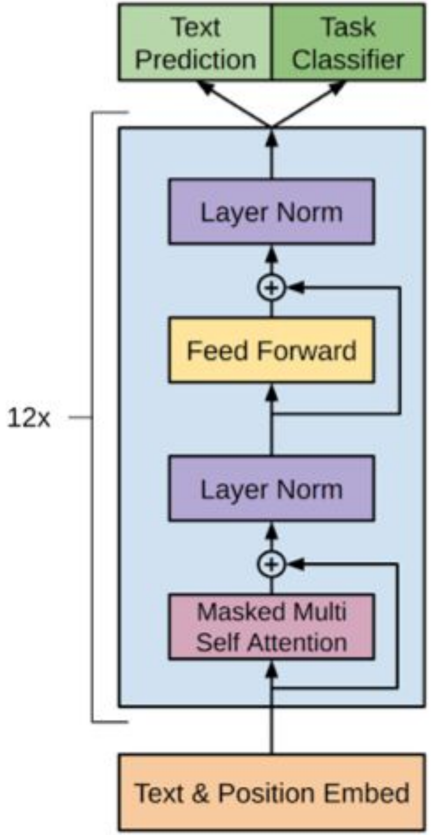


Figure 1: The Transformer - model architecture.

GPT Framework

- Multi-layer Transformer decoder

$$h_0 = UW_e + W_p$$
$$h_l = \texttt{transformer\_block}(h_{l-1}) \forall i \in [1, n]$$



Unsupervised Pre-Training

- Similar objective as Word2Vec
- Given tokens u, maximize:

$$L_1(\mathcal{U}) = \sum_i \log P(u_i | u_{i-k}, \ldots, u_{i-1}; \Theta)$$

$$P(u) = \texttt{softmax}(h_n W_e^T)$$

- k is context window size
- $(h_n W_e^T)$ is score for each word
- softmax gives probability distribution

Supervised Fine-Tuning

- **Keep the pre-trained Transformers**
- Replace the final linear layer
    - Replace W_e with W_y
- Data inputs x, label y
- Maximize

$$L_2(\mathcal{C}) = \sum_{(x,y)} \log P(y|x^1, \ldots, x^m).$$

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

$$L_3(\mathcal{C}) = L_2(\mathcal{C}) + \lambda * L_1(\mathcal{C})$$

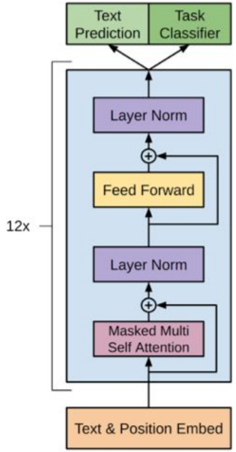
Auxiliary Training Objective

Framework

- Multi-layer Transformer decoder

$$h_0 = UW_e + W_p$$

$$h_l = \texttt{transformer\_block}(h_{l-1}) \forall i \in [1, n]$$



- Unsupervised pre-training
  - Similar objective as Word2Vec
  - Maximize:

$$L_1(\mathcal{U}) = \sum_i \log P(u_i | u_{i-k}, \ldots, u_{i-1}; \Theta)$$

$$P(u) = \texttt{softmax}(h_n W_e^T)$$

- Supervised fine-tuning
  - Data inputs x, label y
  - Maximize

$$L_2(\mathcal{C}) = \sum_{(x,y)} \log P(y | x^1, \ldots, x^m).$$

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

$$L_3(\mathcal{C}) = L_2(\mathcal{C}) + \lambda * L_1(\mathcal{C})$$

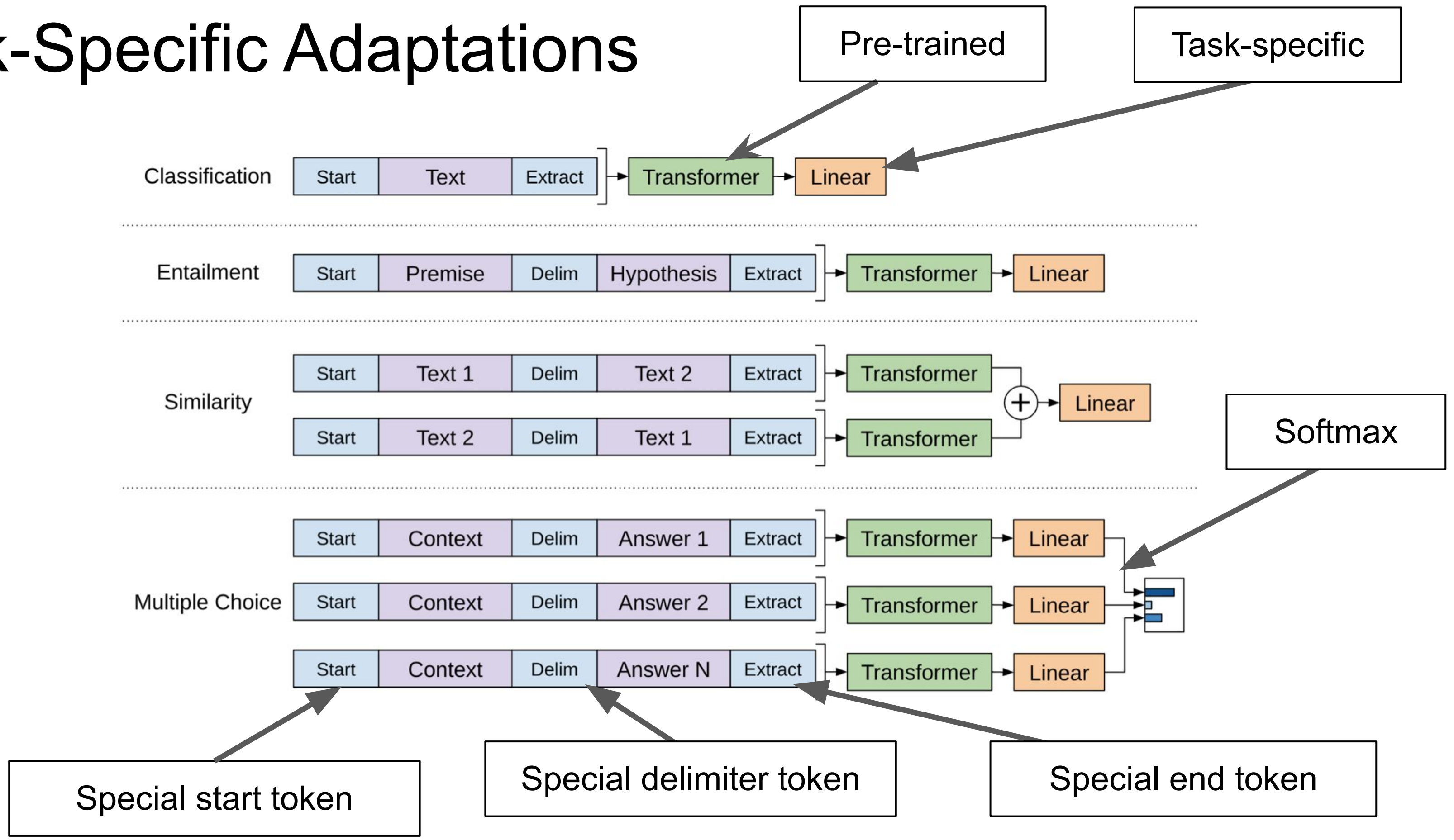
Task Adaptations

- How to adapt a single architecture to multiple input formats?

Task Overviews

- Classification (e.g. sentiment analysis)
  - Given a piece of text, is it positive or negative?
  - Answers: "Yes", "No"
  - Answers: "Very positive", "Positive", "Neutral", "Negative", "Very negative"
- Entailment
  - Given a premise **p** and a hypothesis **h**, does **p** imply **h**?
  - Answers: "entailment", "contradiction", or "neutral"
- Similarity
  - Are two sentences semantically equivalent?
  - Answers: "Yes", "No"
- Multiple Choice (e.g. Story Cloze)
  - Given a short story and two sentences, which is the sentence that ends the story?
  - Given a passage and a question, and some multiple-choice answers, which is the answer?
  - Answers: A_1, A_2, … A_N

Task-Specific Adaptations



Experiments

- BooksCorpus for unsupervised training
    - About the same size as 1B Word Benchmark (used for ELMo)
    - **Preserves longer structure**
- Model
    - 12-layer transformer network
- Returns strong results on most tasks, especially question answering and commonsense reasoning

Data

Textual Entailment: ~2-6% improvement

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

Data

Question answering and story completion: 3-6% improvement

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**

Data

Semantic similarity and text classification: wide range

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**

Takeaways (discussion?)

Takeaways (discussion?)

- **Few new parameters** for each supervised task
  - One linear layer, plus delimiter embedding
- Transformers
  - Allow long-term dependencies to be made
  - Faster to train
- BPE Tokens (next)
- Zero-shot Behavior (next next)

Binary Pair Encoding (BPE) Tokens

- Drawbacks of regular word tokens
  - Other forms? (play vs. playing)
  - Compound words (overripe)
  - Large vocab size
- Begin with a vocabulary: 'A', 'B', 'C', ...
- Add to your vocabulary: Combine common character-pairs
- 'A' + 'B' → 'AB'
- Also, add an end-of-word symbol *
- Example: { 'low', 'lowest', 'newer', 'wider' }
- { 'low*', 'lowest*', 'newer*', 'wider*' }
  - Add { r* , lo , low , er* } to vocabulary
  - Before: l + o + w + e + r + *
  - After: low + er*

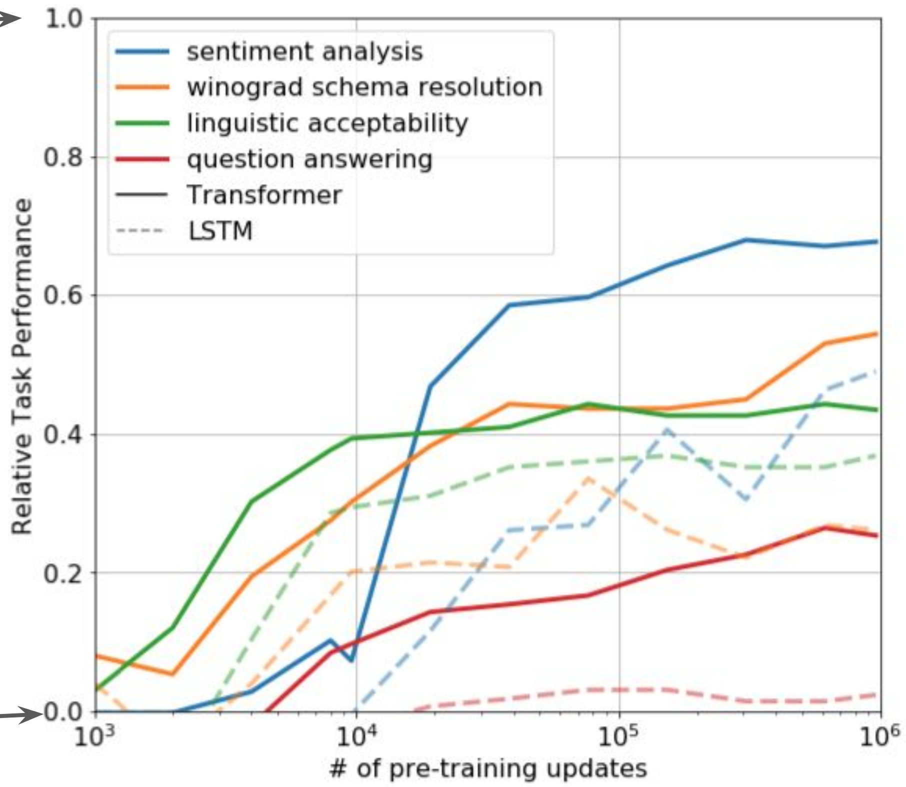
r ·	→	r·
l o	→	lo
lo w	→	low
e r·	→	er·

Zero-shot Behavior

- Use heuristics, rather than supervised training
- Use pre-trained model directly
- E.g: Question answering: Pick the answer the generative model assigns the highest probability to, conditioned on the document and question

Analysis: Zero-Shot Behavior

State of Art

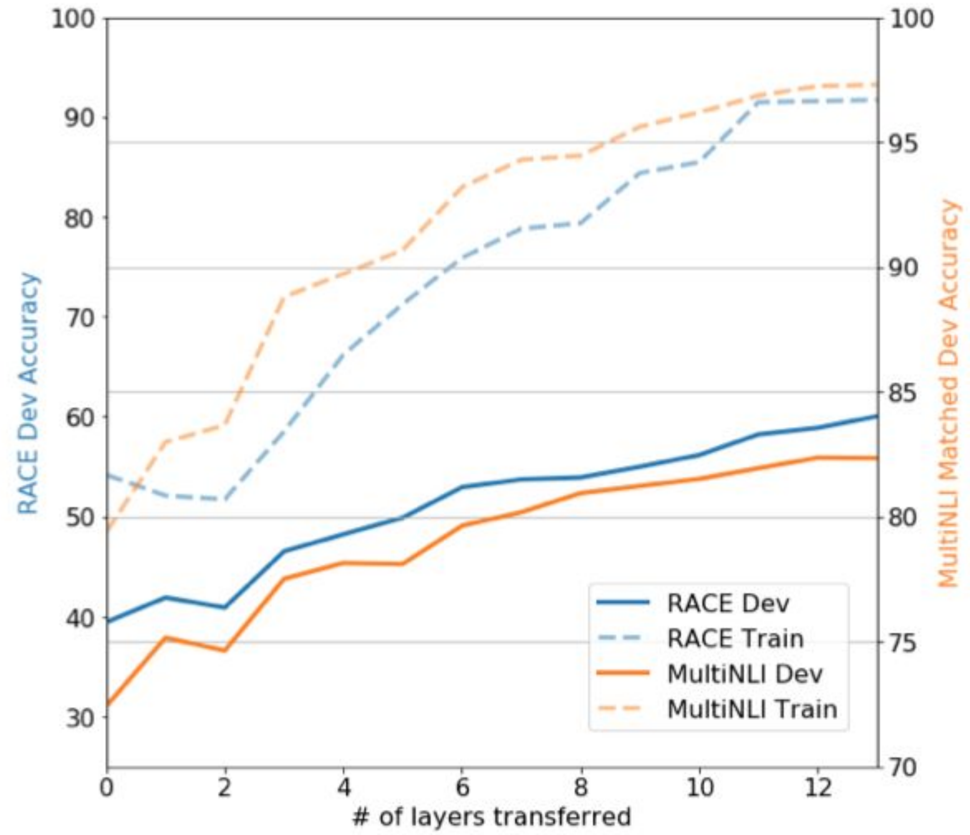


Random Guessing

Takeaways (discussion?)

- **Few new parameters** for each supervised task
  - One linear layer, plus delimiter embedding
- Transformers
  - Allow long-term dependencies to be made
  - Faster to train
- BPE Tokens (next)
- Zero-shot training (next next)

Analysis: Layer Transfer



Related Work

- **Pre-trained LSTM**
  - (Dai et al. 2015) and (Howard and Ruder 2018)
  - Pre-train LSTM's on sequence autoencoding, then apply to text classification
- Auxiliary unsupervised objectives
  - Add an unsupervised goal to your objective
    - E.g. Ask your model to do context-prediction *and* text classification
    - (Collobert and Weston 2008) and (Rei 2017)

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Devlin et al. (Google AILanguage)

Slides From
https://www.slideshare.net/minhpqn/bert-pretraining-of-deep-bidirectionaltransformers-for-languageunderstanding-126429863/37

Previous Work

- Language model pre-training has been used to improve many NLP tasks
  - Elmo(Peters et al., 2018)
  - OpenAI GPT(Radford et al., 2018)
  - ULMFit(Howard and Rudder, 2018)
- Language model pre-training has been used to improve many NLP tasks
  - Feature-based: include pre-trained representations as additional features(e.g., ELMo)
  - Fine-tuning: introduce task-specific parameters and fine-tune the pre-trained parameters

Limitations of Previous Techniques

- **Problem:** Language models only use left context or right context, but language understanding is bidirectional.
- Why are LMs unidirectional?
- Reason: Words can “see themselves” in a bidirectional encoder.

Main Ideas

- Propose a new training objective so that a deep bidirectional transformer can be trained
  - The masked language model
  - Next Sentence Prediction
- Merits of BERT
  - Just fine-tune BERT Model for specific tasks to achieve state-of-the-art performance
  - BERT advances the state-of-the-art for eleven NLP tasks

Masked LM

- **Solution**: Mask out *k*% of the input words, and then predict the masked words
  - We always use $k = 15\%$

```
                store              gallon
                  ↑                  ↑
the man went to the [MASK] to buy a [MASK] of milk
```

- Too little masking: Too expensive to train
- Too much masking: Not enough context

Borrowed From Jacob Devlin's Slides

Masked LM

- Problem: Mask token never seen at fine-tuning
- Solution: 15% of the words to predict, but don't replace with [MASK] 100% of the time. Instead:
- 80% of the time, replace with [MASK]
- 10% of the time, replace random word
- 10% of the time, keep same

Next Sentence Prediction

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

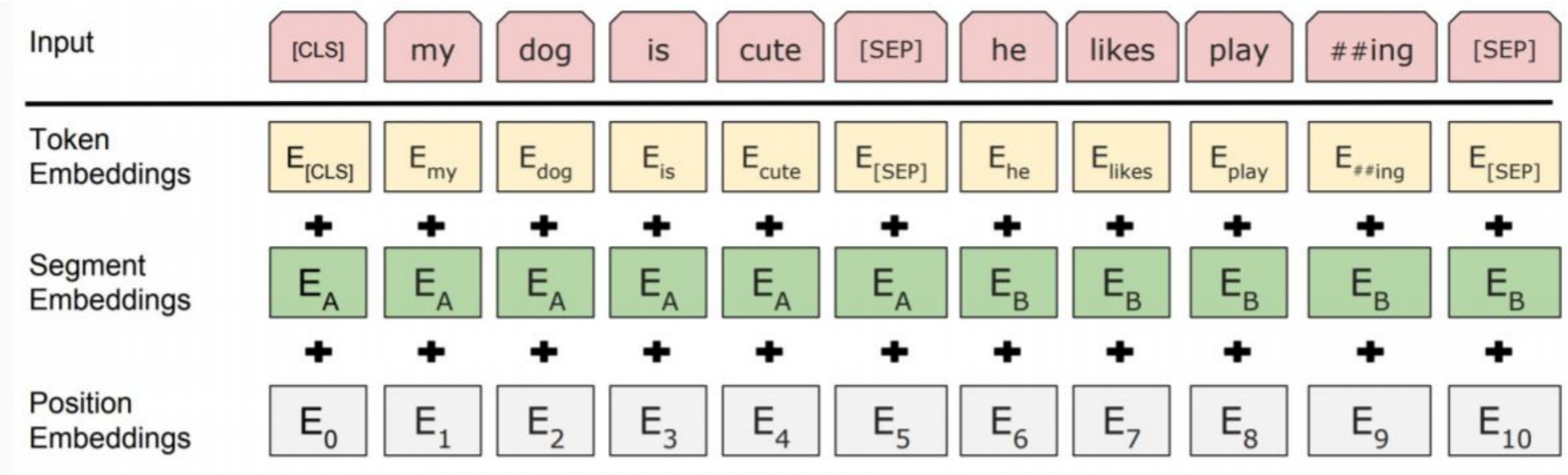
Sentence A = `The man went to the store.`
Sentence B = `He bought a gallon of milk.`
Label = `IsNextSentence`

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

Training Loss

- The training loss is the sum of the mean masked Language Model likelihood and the mean next sentence prediction likelihood

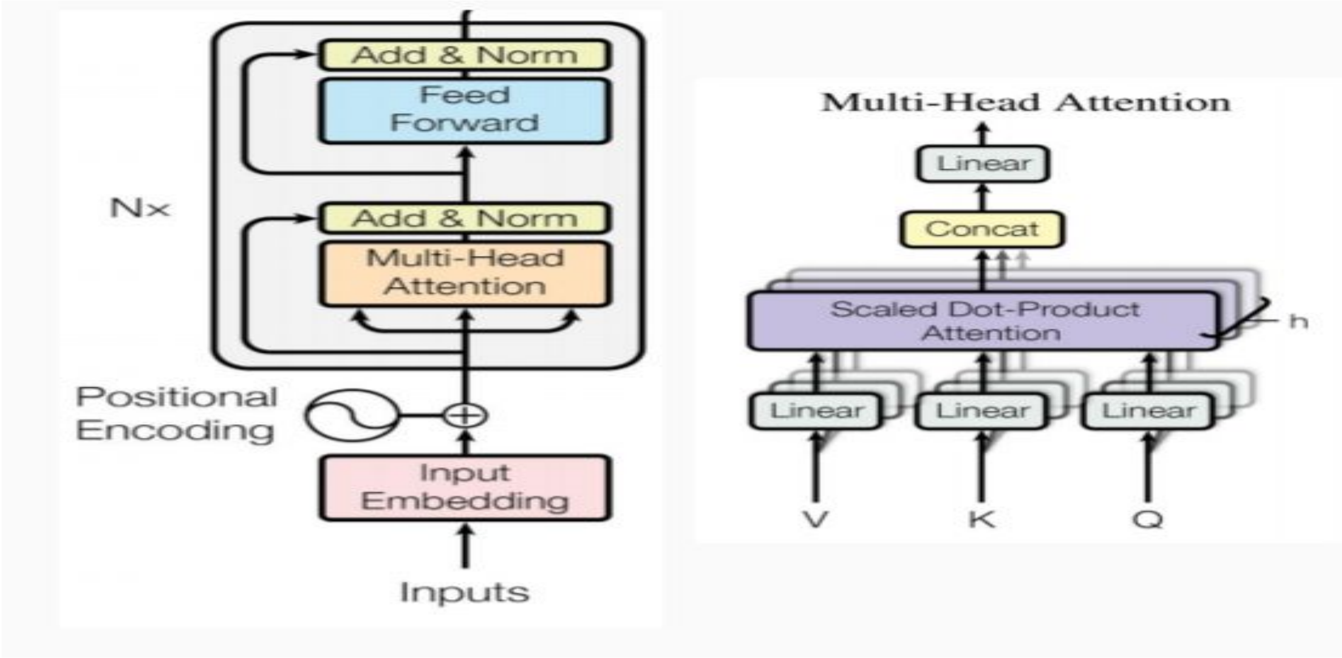
Input Representation



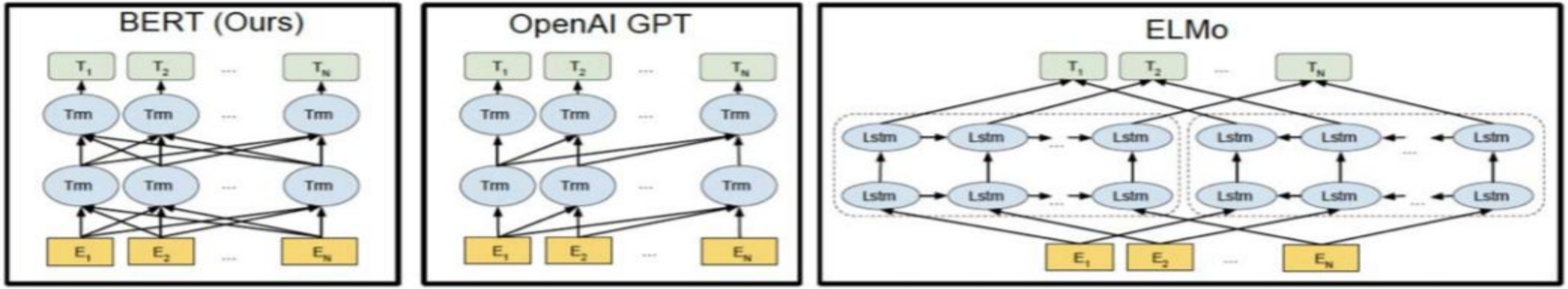
- Use 30,000 WordPiece vocabulary on input
- Each token is sum of three embeddings

Model Architecture

Transformer encoder



Model Architecture



Differences in pre-training model architectures: BERT, OpenAI GPT, and ELMo

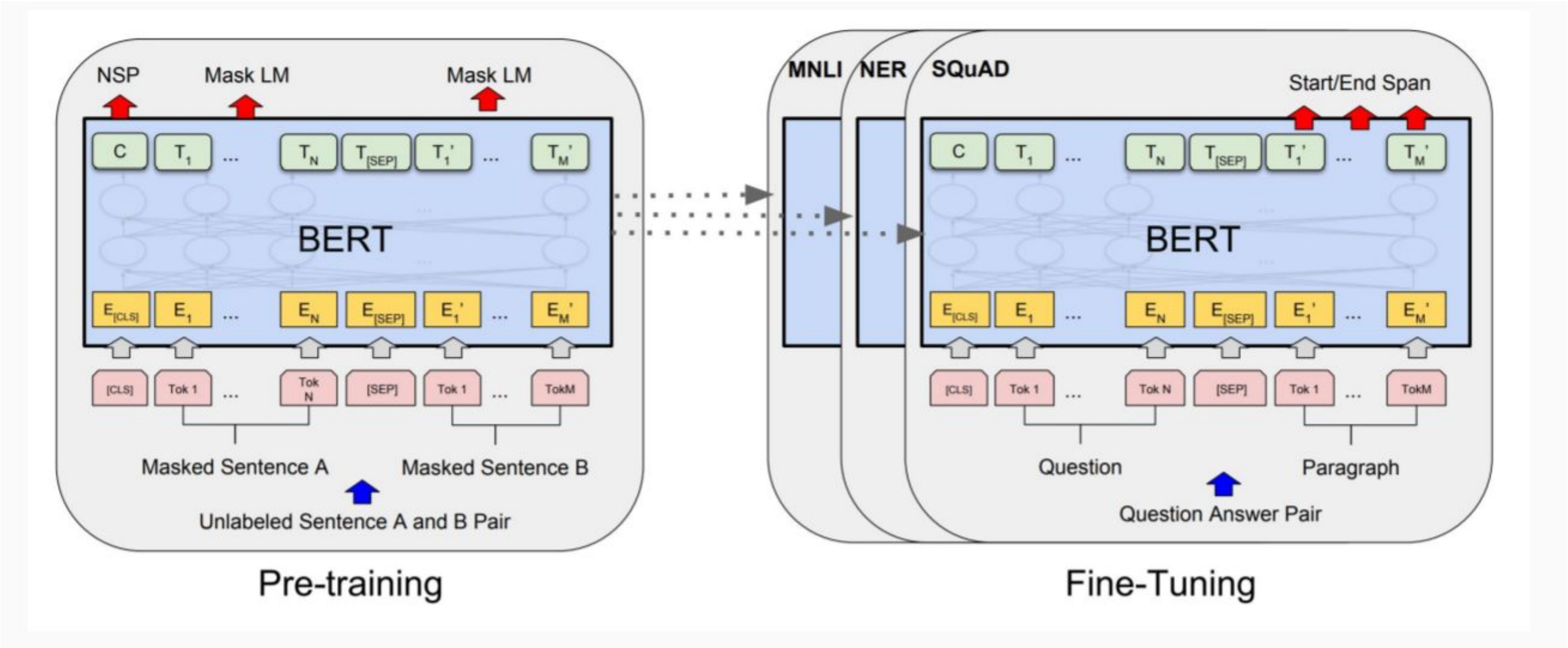
Model Details

- 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
- *BERT-Base*: 12-layer, 768-hidden, 12-head
- *BERT-Large*: 24-layer, 1024-hidden, 16-head
- Trained on 4x4 or 8x8 TPU slice for 4 days

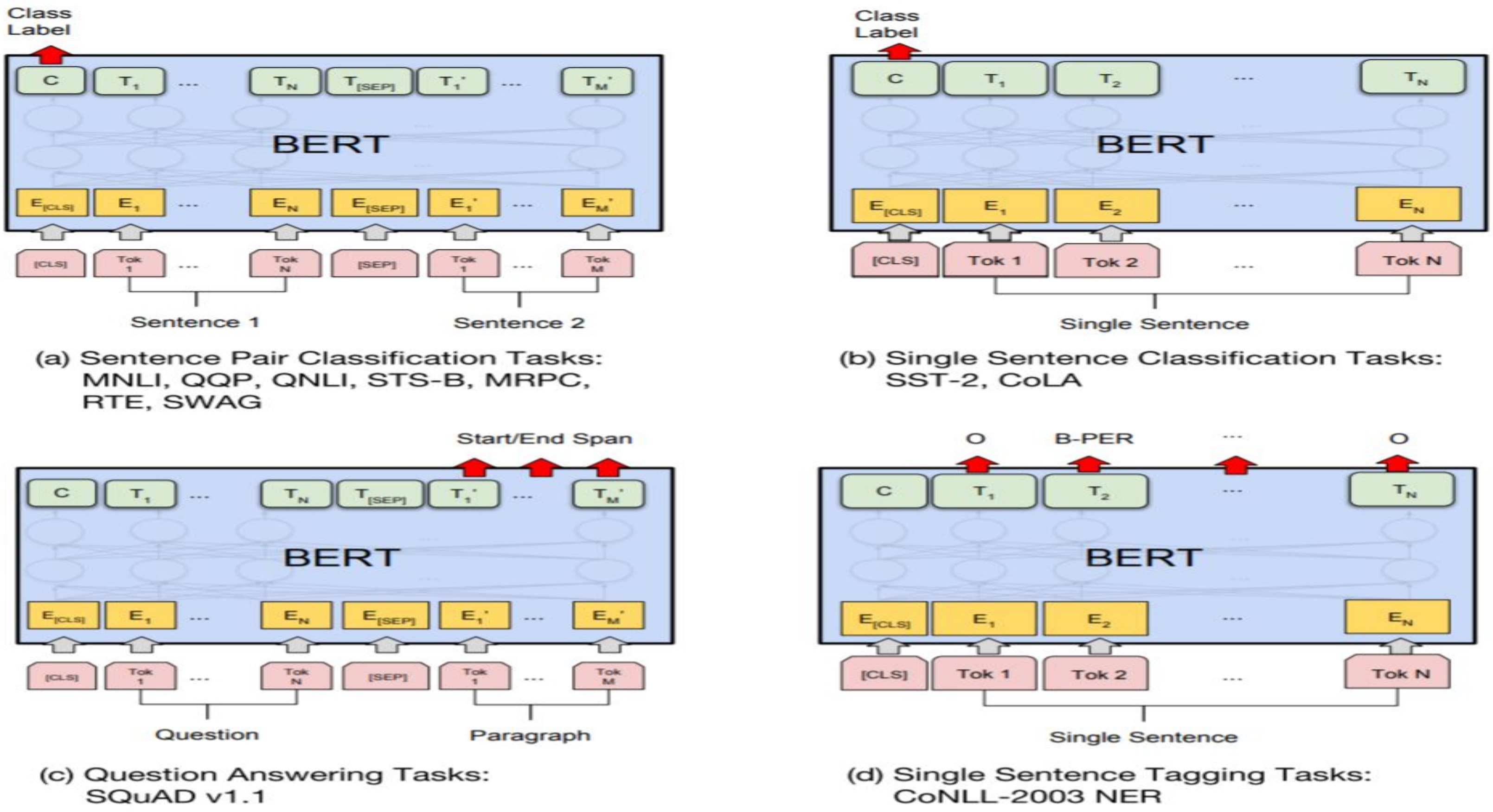
Comparison of BERT and OpenAI GPT

OpenAI GPT	BERT
Trained on BooksCorpus (800M)	Trained on BooksCorpus (800M) + Wikipedia (2,500M)
Use sentence separater ([SEP]) and classifier token ([CLS]) only at fine-tuning time	BERT learns [SEP], [CLS] and sentence A/B embeddings during pre-training
Trained for 1M steps with a batch-size of 32,000 words	Trained for 1M steps with a batch-size of 128,000 words
Use the same learning rate of 5e-5 for all fine-tuning experiments	BERT choose a task-specific learning rate which performs the best on the development set

Fine-Tuning Procedure



Fine-Tuning For Specific Tasks



Results

GLUE Results

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	79.9	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**

MultiNLI

<u>Premise</u>: Hills and mountains are especially sanctified in Jainism.
<u>Hypothesis</u>: Jainism hates nature.
<u>Label</u>: Contradiction

CoLa

<u>Sentence</u>: The wagon rumbled down the road.
<u>Label</u>: Acceptable

<u>Sentence</u>: The car honked down the road.
<u>Label</u>: Unacceptable

SQuAD 1.1

What was another term used for the oil crisis?
Ground Truth Answers: first oil shock shock shock first oil shock shock
Prediction: shock

The 1973 oil crisis began in October 1973 when the members of the Organization of Arab Petroleum Exporting Countries (OAPEC, consisting of the Arab members of OPEC plus Egypt and Syria) proclaimed an oil embargo. By the end of the embargo in March 1974, the price of oil had risen from US$3 per barrel to nearly $12 globally; US prices were significantly higher. The embargo caused an oil crisis, or "shock", with many short- and long-term effects on global politics and the global economy. It was later called the "first oil shock", followed by the 1979 oil crisis, termed the "second oil shock."

- Only new parameters: Start vector and end vector.
- Softr itions.

$$P_i = \frac{e^{S \cdot T_i}}{\sum_j e^{S \cdot T_j}}$$

Rank	Model	EM	F1
	Human Performance *Stanford University* (Rajpurkar et al. '16)	82.304	91.221
1 Oct 05, 2018	BERT (ensemble) *Google AI Language* https://arxiv.org/abs/1810.04805	**87.433**	**93.160**
2 Oct 05, 2018	BERT (single model) *Google AI Language* https://arxiv.org/abs/1810.04805	85.083	91.835
2 Sep 26, 2018	nlnet (ensemble) *Microsoft Research Asia*	85.954	91.677
5 Sep 09, 2018	nlnet (single model) *Microsoft Research Asia*	83.468	90.133
3 Jul 11, 2018	QANet (ensemble) *Google Brain & CMU*	84.454	90.490

The training objective is the sum of the log-likelihoods of the correct start and end positions.

SWAG

SWAG

- The Situations with Adversarial Generations (SWAG)

```
A girl is going across a set of monkey bars.  She
(i) jumps up across the monkey bars.
(ii) struggles onto the bars to grab her head.
(iii) gets to the end and stands on a wooden plank.
(iv) jumps up and does a back flip.
```

- The only task-specific parameters is a vector $V \in \mathbb{R}^H$
- The probability distribution is the softmax over the four choices

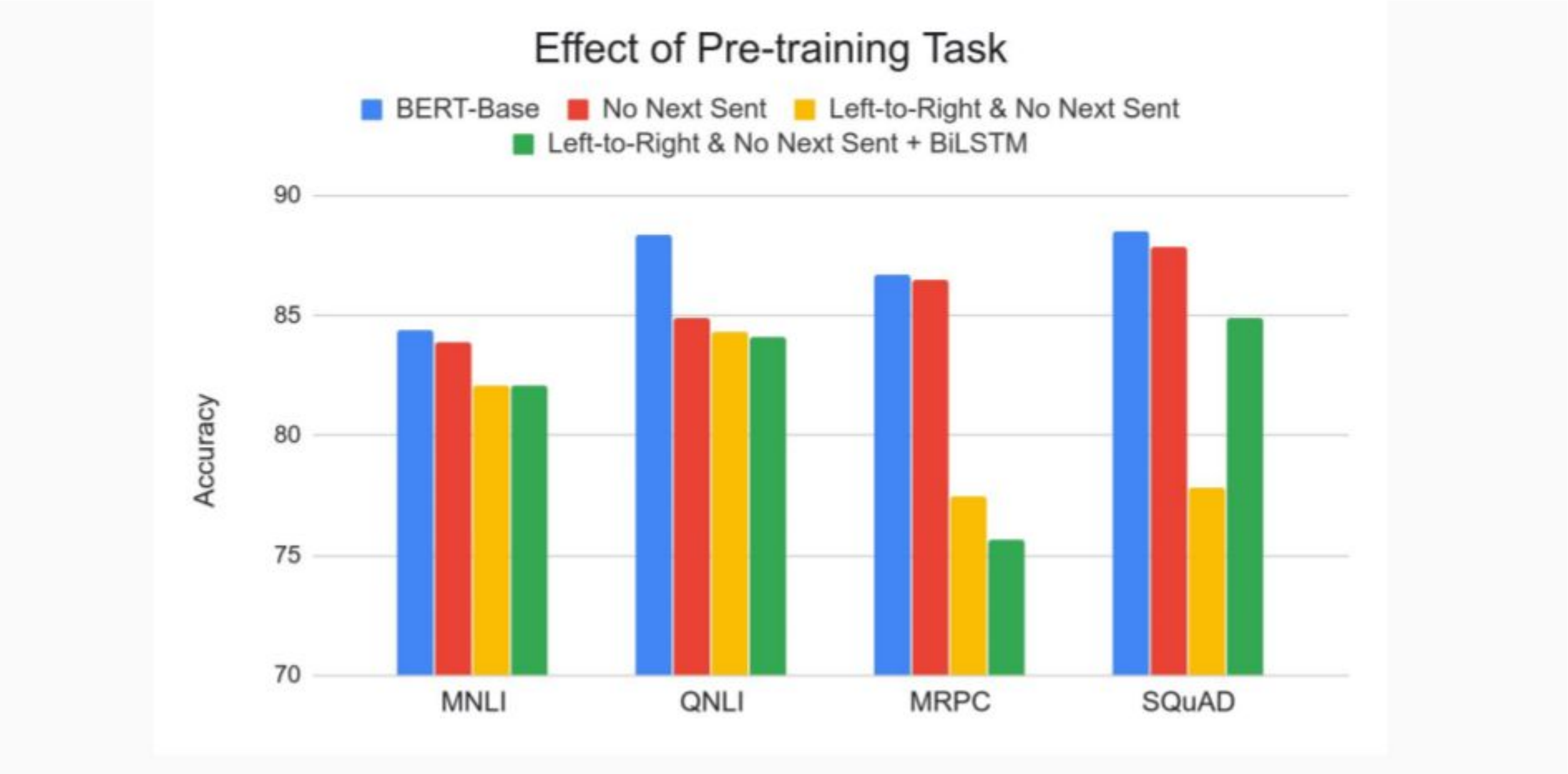
$$P_i = \frac{e^{V \cdot C_i}}{\sum_{j=1}^{4} e^{V \cdot C_i}}$$

Swag Result

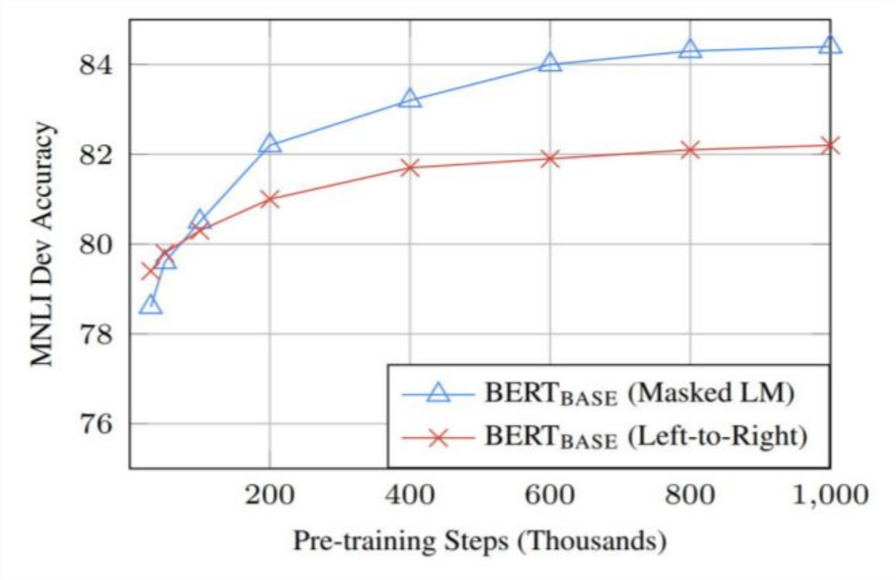
System	Dev	Test
ESIM+GloVe	51.9	52.7
ESIM+ELMo	59.1	59.2
$\text{BERT}_{\text{BASE}}$	81.6	-
$\text{BERT}_{\text{LARGE}}$	**86.6**	**86.3**
Human (expert)†	-	85.0
Human (5 annotations)†	-	88.0

Table 4: SWAG Dev and Test accuracies. Test results were scored against the hidden labels by the SWAG authors. †Human performance is measure with 100 samples, as reported in the SWAG paper.

Ablation Study



Effect of Training Time



- Masked LM takes slightly longer to converge because we only predict 15% instead of 100%
- But absolute results are much better almost immediately

Effects of Model Size

- Big models help *a lot*
- Going from 110M -> 340M params helps even on datasets with 3,600 labeled examples

Effects of Masking Strategy

Masking Rates			Dev Set Results		
MASK	SAME	RND	MNLI Fine-tune	NER Fine-tune	NER Feature-based
80%	10%	10%	84.2	95.4	94.9
100%	0%	0%	84.3	94.9	94.0
80%	0%	20%	84.1	95.2	94.6
80%	20%	0%	84.4	95.2	94.7
0%	20%	80%	83.7	94.8	94.6
0%	0%	100%	83.6	94.9	94.6

- Masking 100% of the time hurts on feature-based approach

- Using random word 100% of time hurts slightly

Feature-Based Approach Using BERT

Advantages of Feature-Based Approach:

- Not all tasks can be represented by a transformer encoder architecture, and therefore require a task-specific model architecture to be added
- Major computational benefits to pre-compute an expensive representation of the training data once and then run many experiments with cheaper models on top of this representation.

Feature-Based BERT Results

System	Dev F1	Test F1
ELMo (Peters et al., 2018a)	95.7	92.2
CVT (Clark et al., 2018)	-	92.6
CSE (Akbik et al., 2018)	-	**93.1**
Fine-tuning approach		
$\text{BERT}_{\text{LARGE}}$	96.6	92.8
$\text{BERT}_{\text{BASE}}$	96.4	92.4
Feature-based approach ($\text{BERT}_{\text{BASE}}$)		
Embeddings	91.0	-
Second-to-Last Hidden	95.6	-
Last Hidden	94.9	-
Weighted Sum Last Four Hidden	95.9	-
Concat Last Four Hidden	96.1	-
Weighted Sum All 12 Layers	95.5	-

Table 7: CoNLL-2003 Named Entity Recognition results. Hyperparameters were selected using the Dev set. The reported Dev and Test scores are averaged over 5 random restarts using those hyperparameters.

Comparison to Computer Vision

- Take a ConvNet pretrained on ImageNet, remove the last fully-connected layer (this layer’s outputs are the 1000 class scores for a different task like ImageNet), then treat the rest of the ConvNet as a fixed feature extractor for the new dataset.

Conclusions

- Pre-trained language models are increasingly adopted in many NLP tasks
- Major contribution of this paper is to propose a deep bidirectional architecture from Transformer