Transfer Learning in NLP

International Summer School on Deep Learning 2019

Daniel Pressel
Interactions LLC

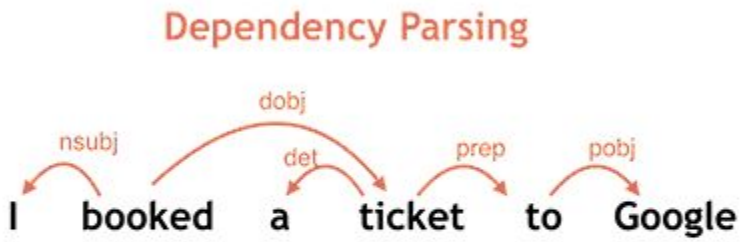
Deep Learning has transformed NLP

- Deep Learning success in NLP: A non-exhaustive list:
  - Named Entity Recognition
  - Part-of-speech Tagging
  - Machine Translation
  - Parsing
  - Document Classification
  - Question Answering
  - Coreference Resolution
  - Recognizing Textual Entailment

Dependency Parsing

- Parse sentence into a graph of arcs between dependents and their heads
- Transition Parsing (e.g. Arc Standard):
  - Define a set of valid moves that used together will yield a parse graph
  - Start with an initial “configuration”
  - At each step, ask a “guide” to predict a transition until we have covered the sentence

Arc-Standard Parsing



* https://ai.googleblog.com/2016/05/announcing-syntaxnet-worlds-most.html

DNNs improve transition parsing!

- Prior to 2014
  - Use SVM or Perceptron as guide
- Chen and Manning 2014:
  - MLP guide
- Kipperwasser and Goldberg 2016
  - BiLSTM + MLP (pictured)
- Stack pointer network from Ma et al., 2018
- Fernandez-Gonzalez and Gomez-Rodriguez, 2019
  - pointer network
  - eliminate stack and buffer
  - only 1 transition type!

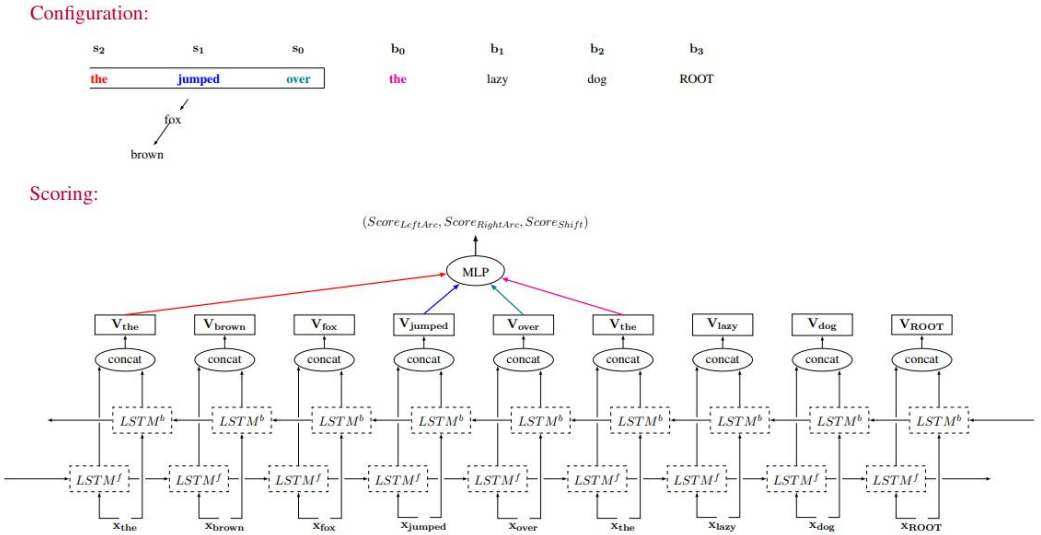


Figure 1: Illustration of the neural model scheme of the transition-based parser when calculating the scores of the possible transitions in a given configuration. The configuration (stack and buffer) is depicted on the top. Each transition is scored using an MLP that is fed the BiLSTM encodings of the first word in the buffer and the three words at the top of the stack (the colors of the words correspond to colors of the MLP inputs above), and a transition is picked greedily. Each x_i is a concatenation of a word and a POS vector, and possibly an additional external embedding vector for the word. The figure depicts a single-layer BiLSTM, while in practice we use two layers. When parsing a sentence, we iteratively compute scores for all possible transitions and apply the best scoring action until the final configuration is reached.

The Changing Landscape of Dependency Parsing

Before Chen and Manning 2014*

Parser	Dev		Test		Speed
	UAS	LAS	UAS	LAS	(sent/s)
standard	90.2	87.8	89.4	87.3	26
eager	89.8	87.4	89.6	87.4	34
Malt:sp	89.8	87.2	89.3	86.9	469
Malt:eager	89.6	86.9	89.4	86.8	448
MSTParser	91.4	88.1	90.7	87.6	10
Our parser	**92.0**	**89.7**	**91.8**	**89.6**	**654**

Table 5: Accuracy and parsing speed on PTB + Stanford dependencies.

After Chen and Manning 2014**

Parser	UAS	LAS
Chen and Manning (2014)	91.8	89.6
Dyer et al. (2015)	93.1	90.9
Weiss et al. (2015)	93.99	92.05
Ballesteros et al. (2016)	93.56	91.42
Kiperwasser and Goldberg (2016)	93.9	91.9
Alberti et al. (2015)	94.23	92.36
Qi and Manning (2017)	94.3	92.2
Fernández-G and Gómez-R (2018)	94.5	92.4
Andor et al. (2016)	94.61	92.79
Ma et al. (2018)*	95.87	94.19
This work*	**96.04**	**94.43**
Kiperwasser and Goldberg (2016)	93.1	91.0
Wang and Chang (2016)	94.08	91.82
Cheng et al. (2016)	94.10	91.49
Kuncoro et al. (2016)	94.26	92.06
Zhang et al. (2017)	94.30	91.95
Ma and Hovy (2017)	94.88	92.96
Dozat and Manning (2016)	95.74	94.08
Ma et al. (2018)*	95.84	94.21

* Chen and Manning, 2014

**Fernandez-Gonzalez and Gomez-Rodriguez, 2019

Transfer Learning has Transformed Deep Learning for NLP!

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	79.8	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.

*Devlin et al. 2019

Named Entity Recognition (NER)

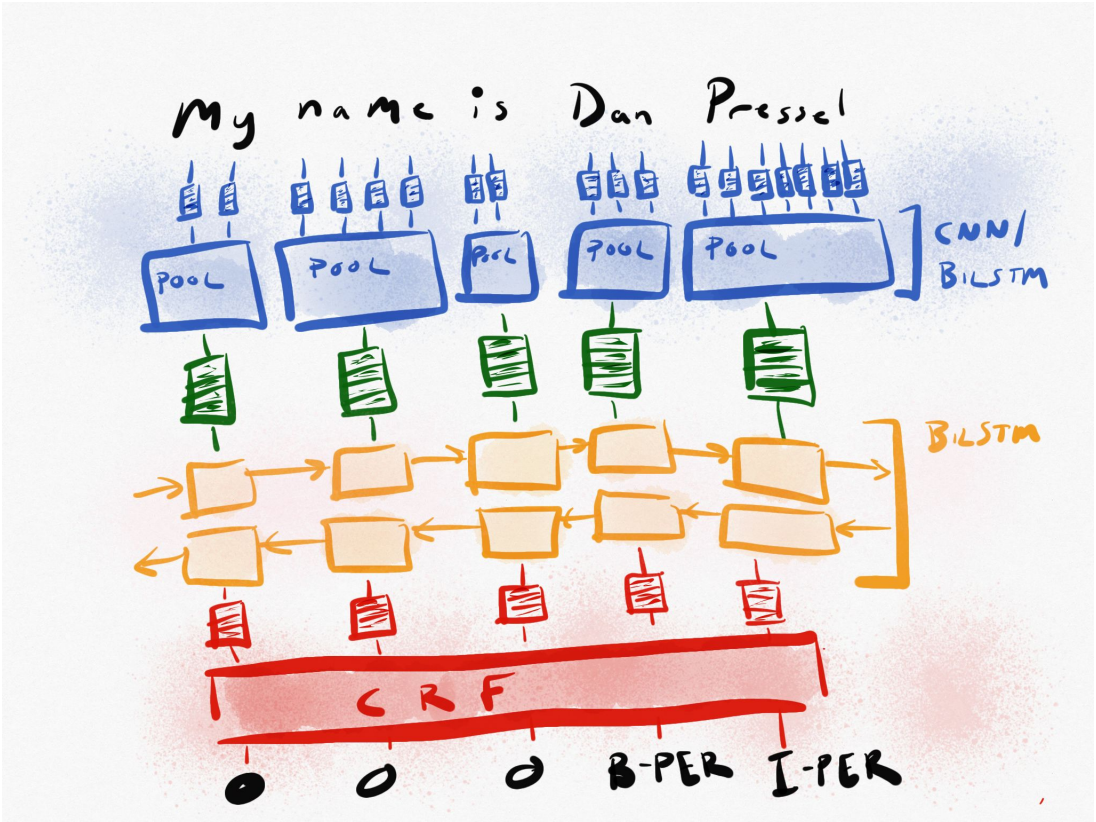
- Named Entity Recognition is a task to spot phrases that are entities and label the entity type

```
My name is Dan   Pressel and I live in the US
O  O    O  B-PER I-PER   O   O O    O  O   B-LOC
```

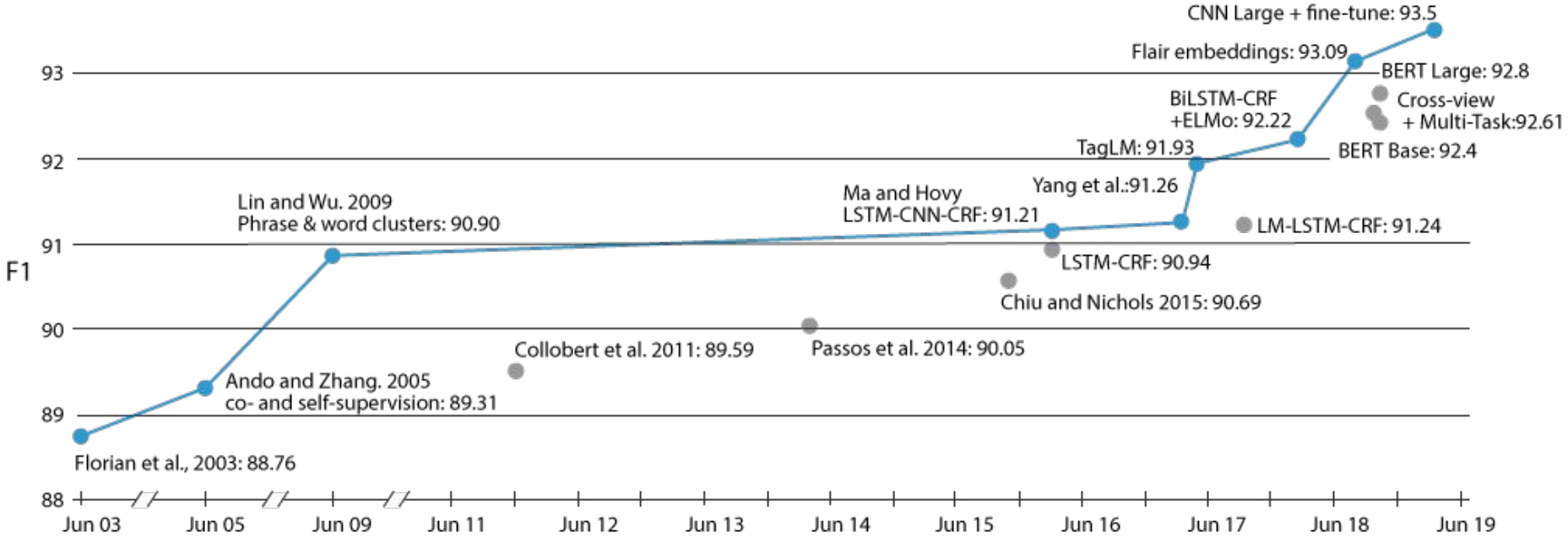
Named Entity Recognition before DNNs (NER)

- Define a set of features to help identify named entities
  - Word shape
  - Gazetteers
- Use a structured classifier that predicts the most likely coverage through a sentence
  - MaxEnt Markov Model (MEMM)
  - Conditional Random Field (CRF)

Named Entity Recognition after DNNs (NER)



DNNs and Transfer Learning are Helping!



*Ruder, Peters, Swayamdipta & Wolf, NAACL, 2019

Sota in nlp 2019

- Many State-of-the-Art models are built using transfer learning
- Most successful technique is generative pre-training of a language model
  - First, learn to predict words
  - Train on a large corpus of text, transfer to downstream application

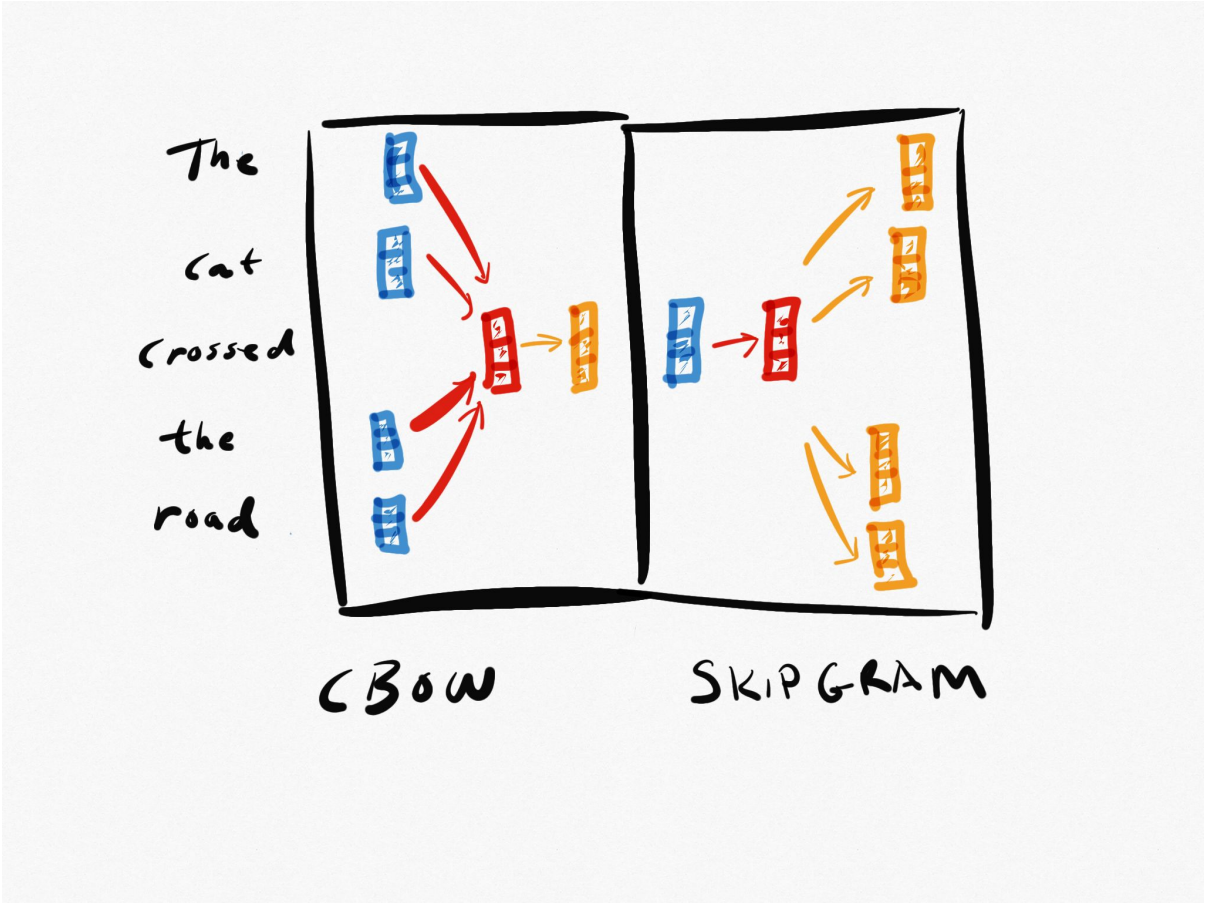
First Some Background

- Use of DNNs has changed the starting point for NLP problems a bit
  - Convert sparse representations to dense continuous ones
- Often use a pre-training technique like **word2vec** to create a distributed representation and plug those in

> “You shall know a word by the company it keeps” - J.R. Firth, 1957

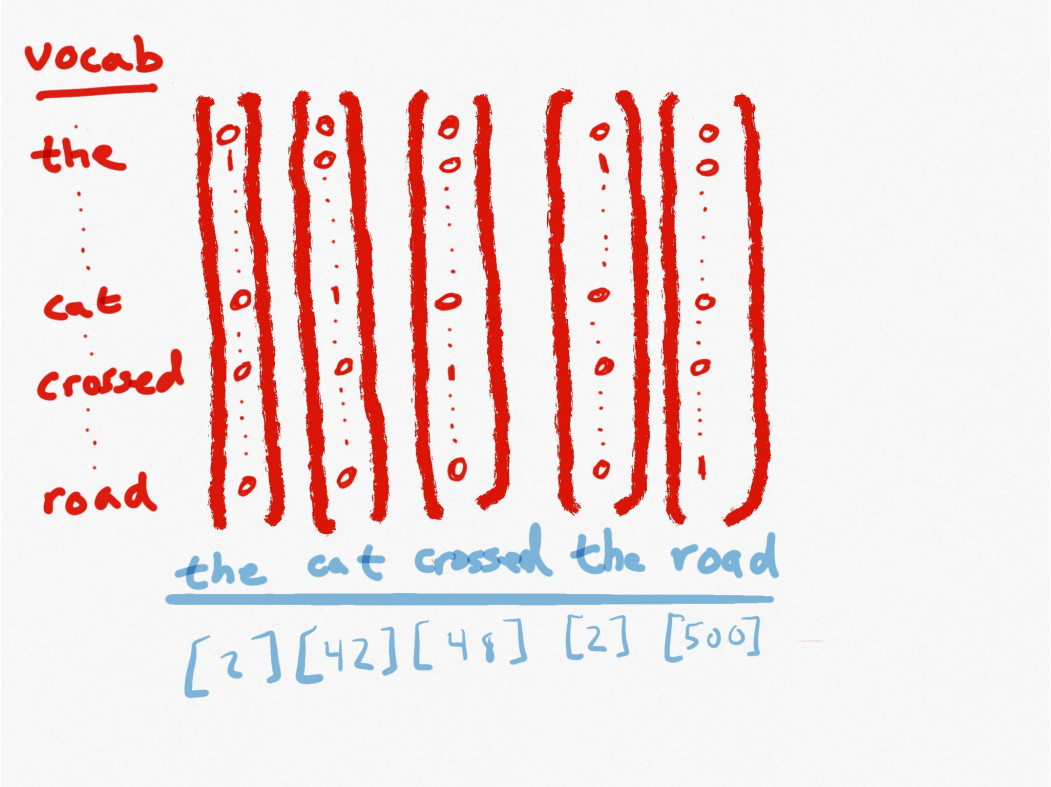
Word2Vec objectives

- CBOW: Given fixed surrounding window context, predict the middle word
- Skip-gram: Given middle word, predict fixed surrounding window



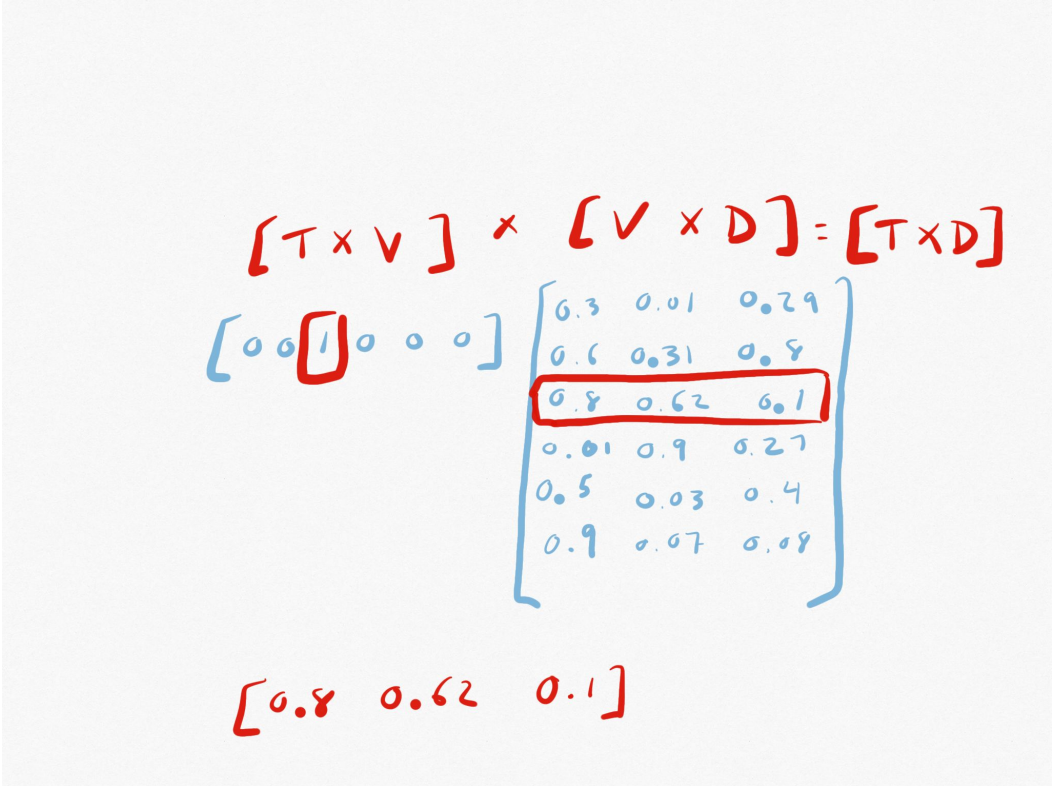
One-hot vectors

- One-hot: with |V| array of vocabulary, only one “on” (1), the rest “off” (0)
- Represents the word at the temporal position t in T
- |T|x|V| array representing a sentence



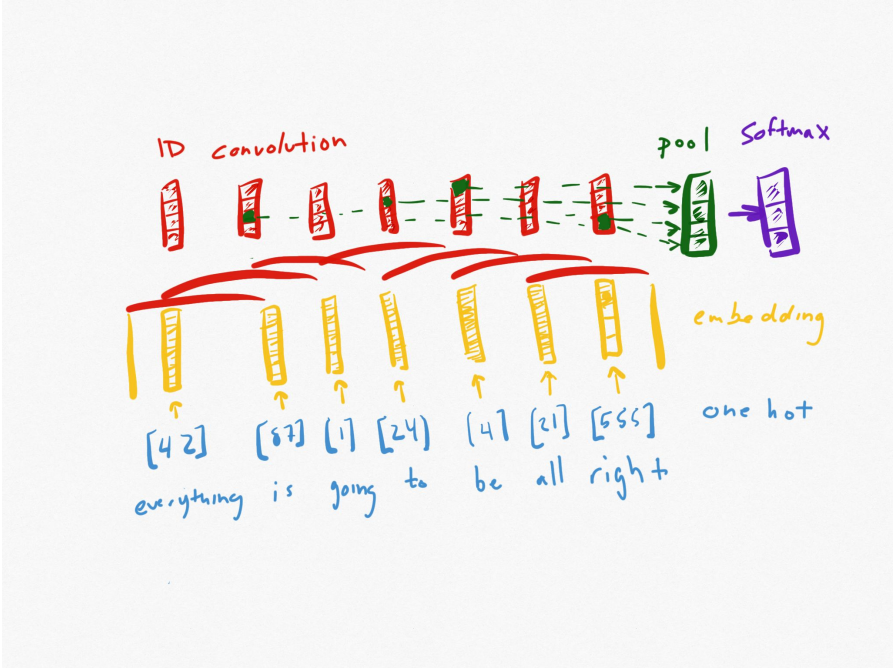
Lookup table-based Word embeddings

- One-hot vector multiply by weight matrix yields row
- Equivalent to looking up by the index
  - Efficient, tensor contains only indices for “on” values



Word embeddings in a classification architecture

- Embeddings make up lowest layer, feed to some pooling mechanism
  - LSTM final hidden state
  - Convolutional Net followed by Max pooling
  - Max/Mean pooling
- Some optional stacking followed by a projection to number of classes



*Collobert et al 2011

Motivation for Contextualized representations

- Pre-trained embeddings caused breakthrough in NLP
  - E.g. Classification and NER started to rely heavily on these features
  - Linear and deep models started to use these features
- For any surface representation, there is only one word vector
  - It seems like the same surface word should have different representations when the context differs
  - How can we learn contextual word vectors?

Causal language modeling

I'd like an Italian sub with everything, light _____ .

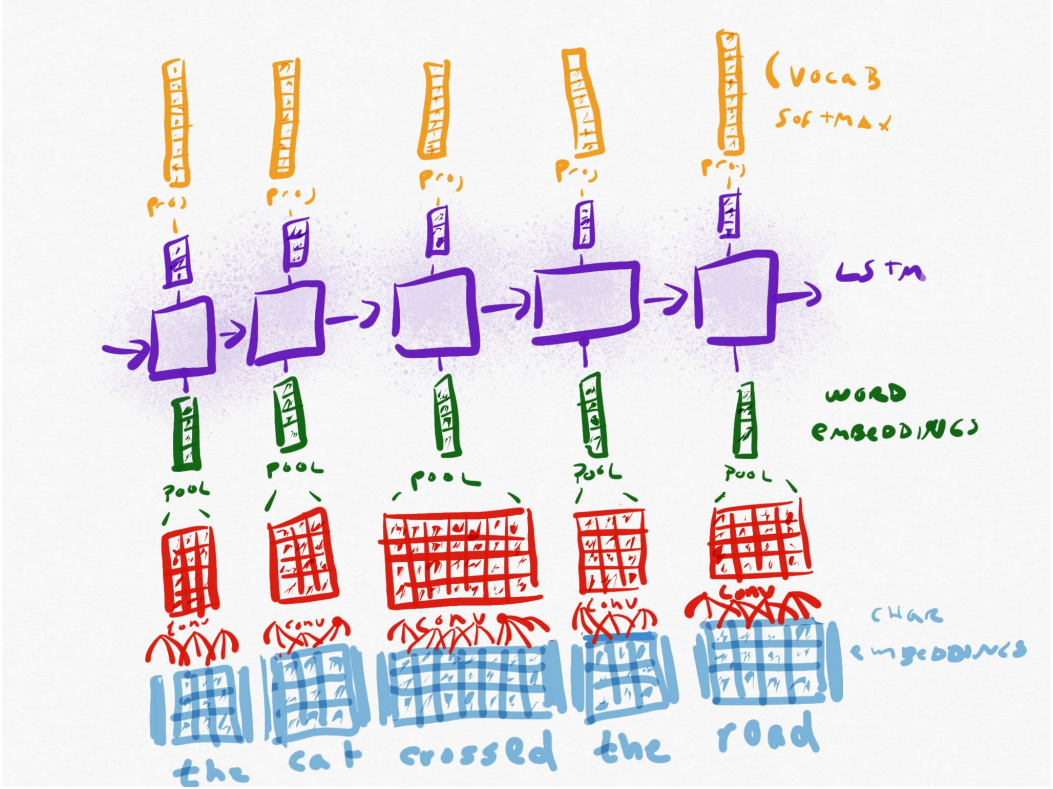
- Can you guess the next word?
  - It probably is not "toothbrush" or "sandbox"
  - Maybe "oil?"
- Can we teach a model to predict it?
  - Intuitively, we'd like a low probability on "toothbrush" and a high probability on "oil"
- IRL, vocabulary is huge
  - How to handle unknown words?

Why Language Models For Pretraining?

- Previous slide foreshadows how difficult this task can be
- Model is forced to learn some syntax, semantics, coreference resolution, dependency parsing to try and solve
- Unlike other tasks we might use, the training data is unlimited

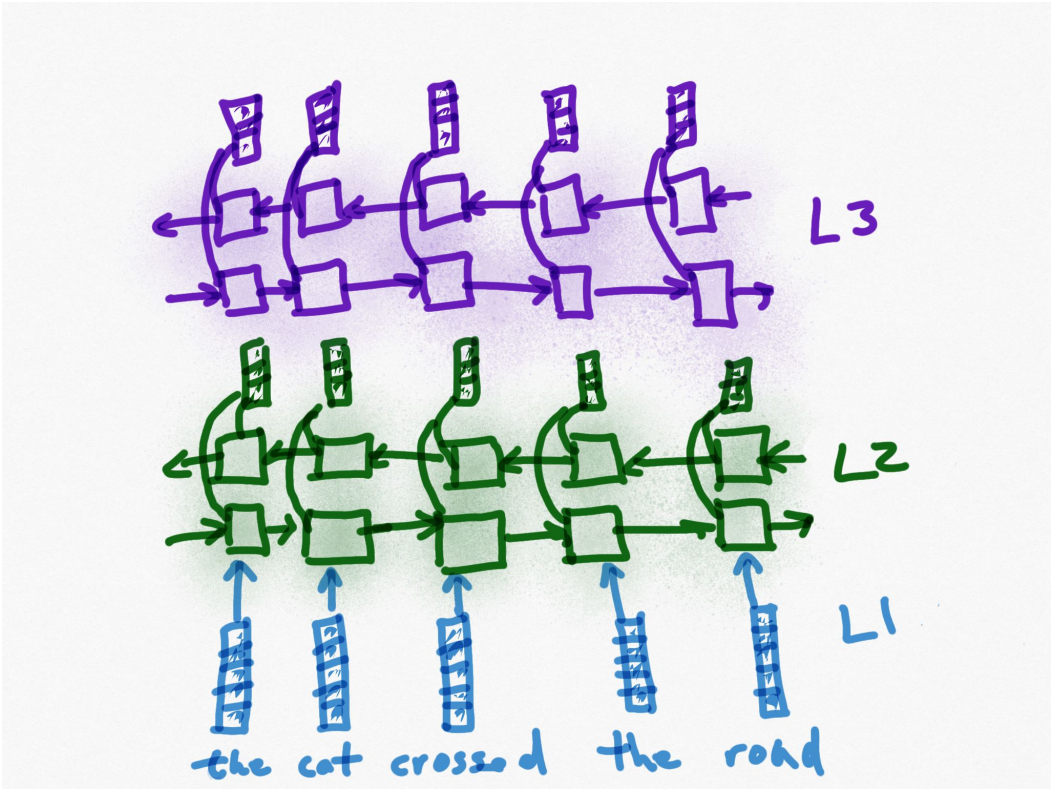
An LSTM language model with characters

- Replace word lookups with char lookups over word
- Convolutional max-over-time pooling
- One or more highway layers
- One or more LSTM layers
- Projection to vocabulary size
- Softmax
- Can train left-to-right and right-to-left and sum losses for biLM



Ok, so we trained a language model, now what?

- ELMo-style biLM encoder
  - Character-word embeddings at layer 1
  - biLSTM layer 2
  - biLSTM layer 3



Some options for downstream use

- Transform each input into a contextualized representation
  - Freeze them or fine-tune? Maybe slow gradients?
  - Pool them and fine-tune the whole model
  - ELMo objective
- According to Peters et al., 2019, use as features when downstream task is very different

LSTM-based LMs

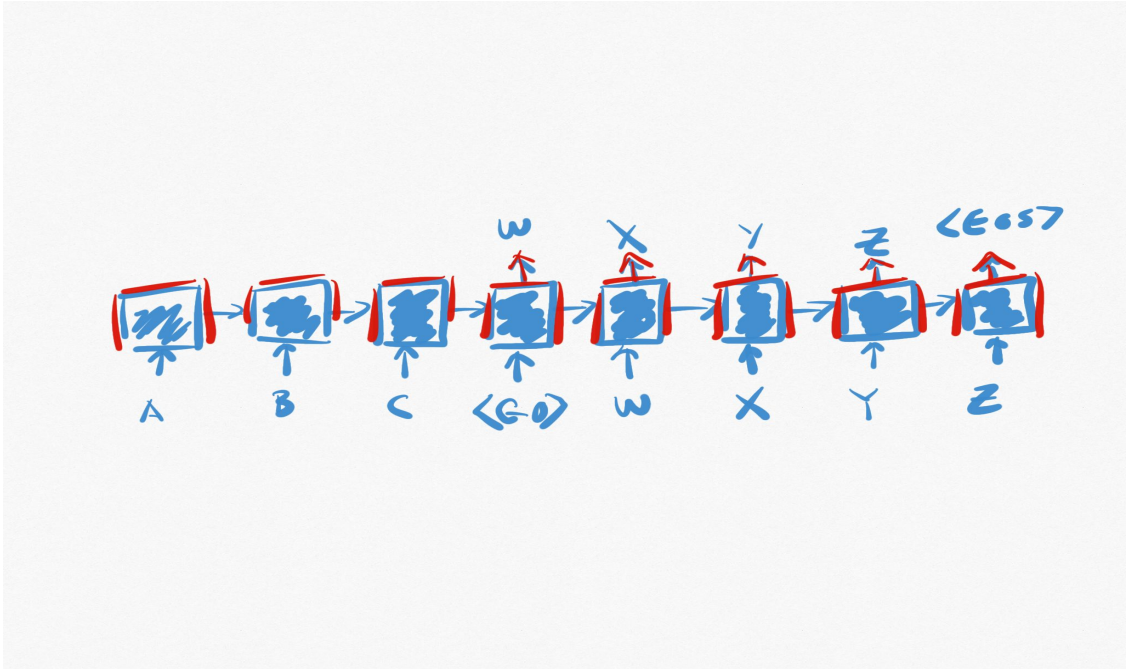
- Learn different representations at different layers, just like in CV
  - As layers get higher, representation moves from syntax to meaning
- Many tasks in NLP and each requires some different degree of knowledge
  - Implies that different contributions desirable for different layers depending on downstream task
  - Train a linear combination of layers

But I heard Attention is all you Need??

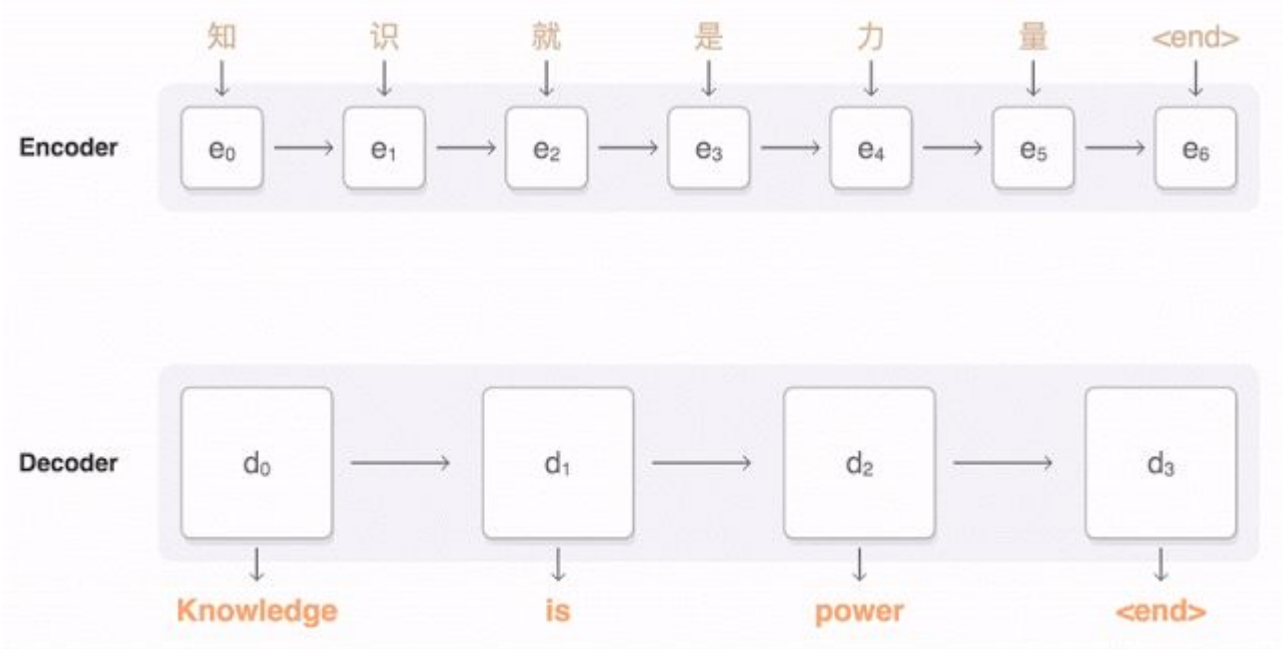
- Goal: eliminate LSTMs
  - Hard to parallelize due to autoregressive nature
  - Even with LSTM, long distance dependencies are challenging
- But LSTMs are shown to be useful for language, how to get around them?
  - Seq2seq already uses attention, can we just do that?

Background: Vanilla Seq2Seq

- Translates but doesn't perform well on long contexts



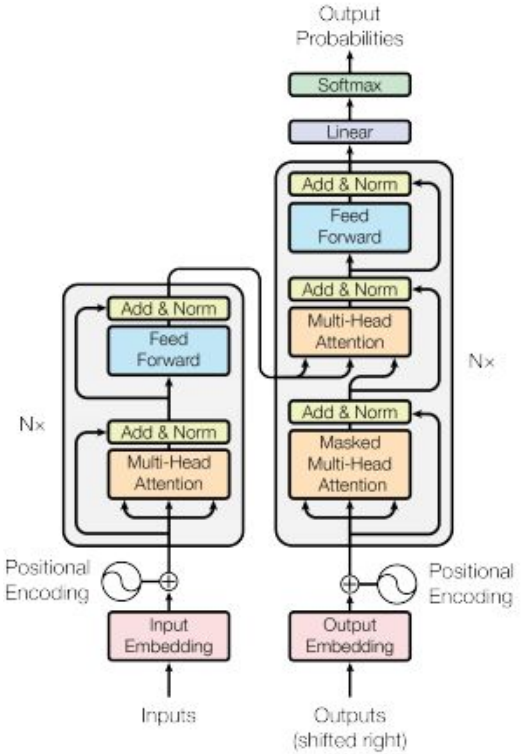
Background: Seq2Seq with Attention



Background: Seq2Seq with Attention

- Linear combination of input informs output token
- Works incredibly well
  - Every seq2seq model today uses attention
  - What if we replace every LSTM with attention?

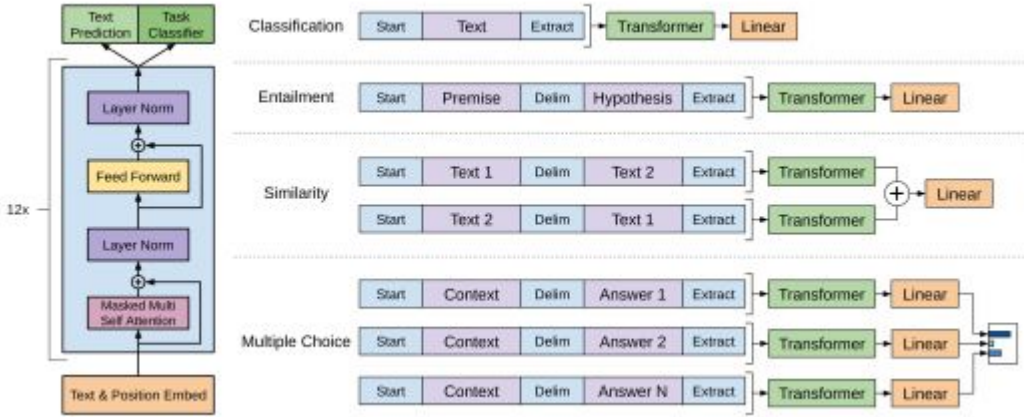
The Transformer



Transformer innovations

- Multi-head attention
- Lots of layer normalization
- Self-attention in encoder and decoder
- Pyramidal mask to mask futures
- Linear warm-up in training regime
- Need some way to distinguish between same word at offset 6 and 14
  - Use positional embeddings

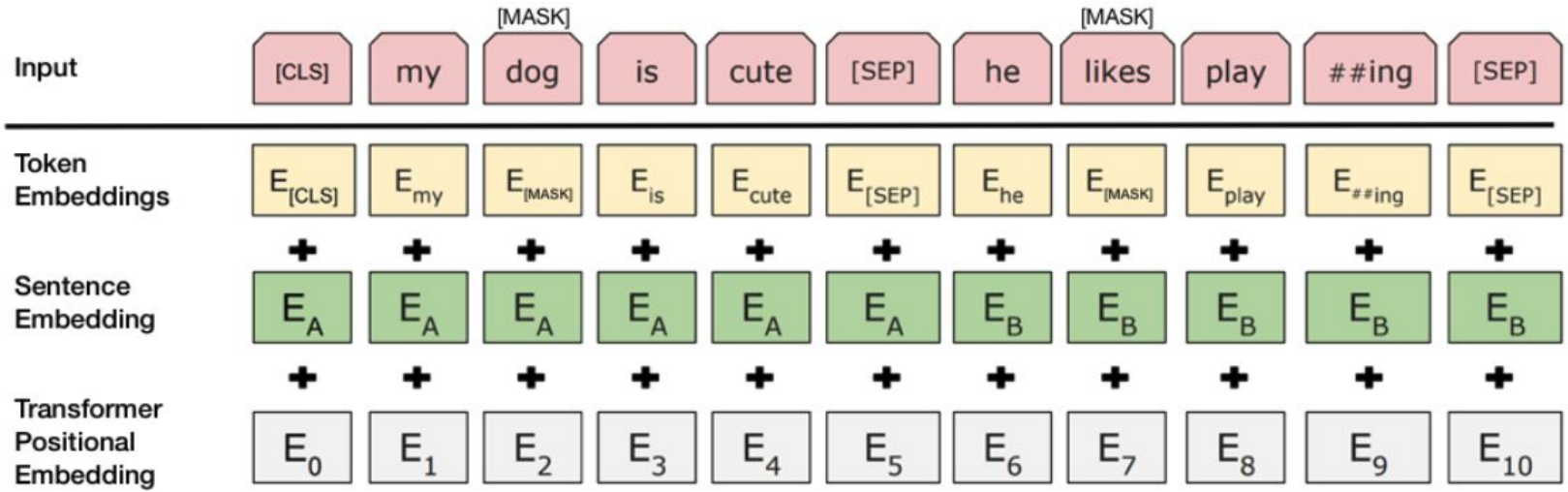
GPT: Transformers are cool! Lets use for Pre-training!



Pre-training Architectures: GPT

- Method
  - Train Causal Transformer Encoder
  - For downstream tasks, remove LM head and replace with downstream head
  - Use BPE instead of character-level modeling
- Strengths
  - Can parallelize, pretty optimal on GPU hardware
  - High capacity pre-trained LM yields strong results on downstream tasks
  - BPE is much faster than character-level modeling
  - Trained on a much larger corpus than ELMo with LDD
  - Large context window (256)
- Weaknesses
  - BPE is not ideal for tasks that need morphological features
  - Unidirectional LM

BERT: GPT is cool but BiLM is important!



Pre-training Architectures: BERT

- Method
  - Train 12-24 Layer Transformer with Next Sentence Prediction (NSP) Task and Masked Language Model (MLM) Task
  - For downstream tasks, remove LM head and replace with downstream head
  - Use BPE instead of character-level modeling
- Strengths
  - Optimized for downstream tasks not LM
    - SoTA on many tasks, researchers are still discovering new strongs
  - BPE is much faster than character-level modeling
  - Trained on a massive corpus
- Weaknesses
  - Subword not ideal for tasks that need morphological features
  - Cannot easily compare LM performance
  - MLM objective is slow to train

GPT-2: No seriously, GPT IS cool

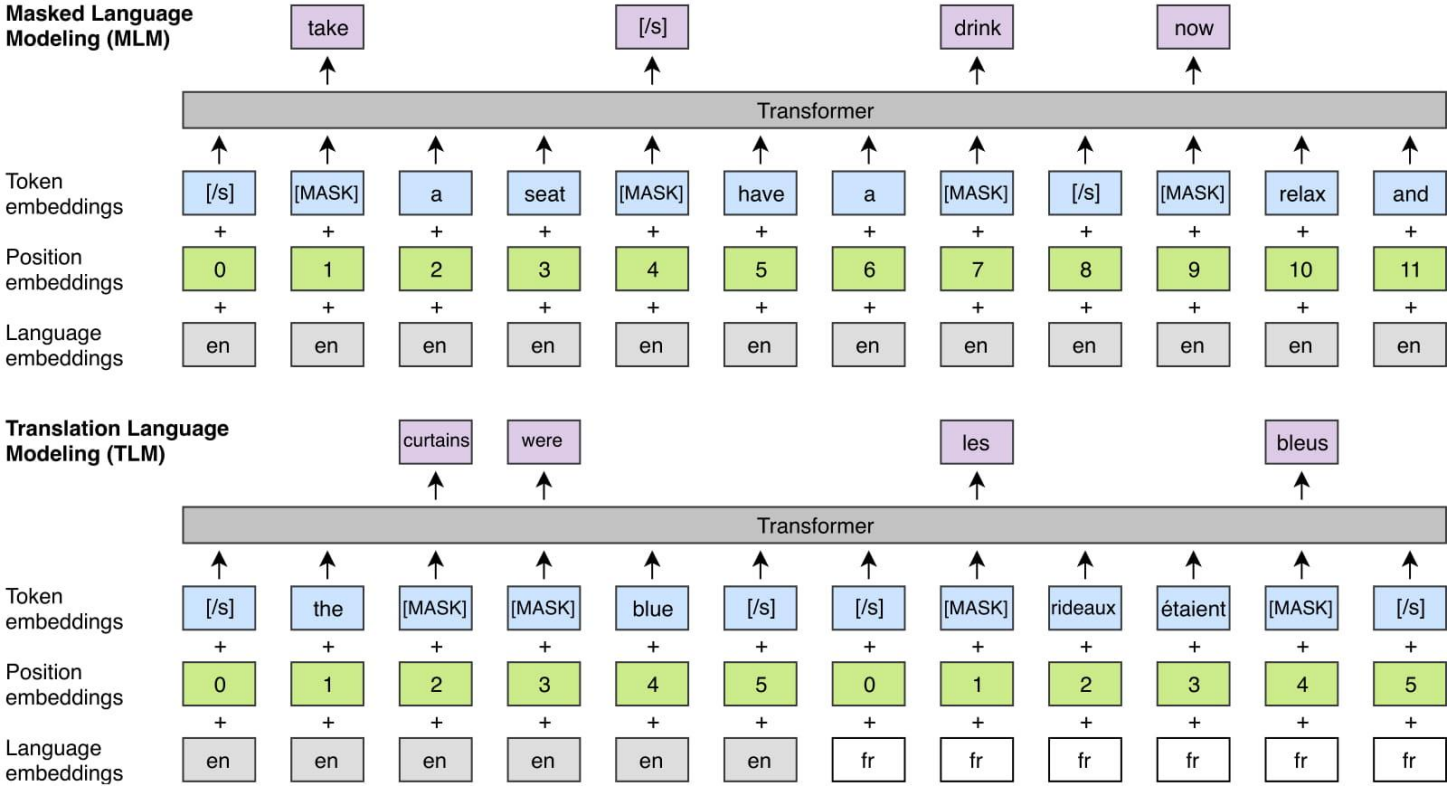
- Scale up GPT
  - Massive context (1024)
  - Larger vocab (~50k)
  - Moves layer norm around
  - Changes initialization
- Zero-shot gets SoTA on well-studied datasets!
- Generates long, relatively coherent statements

GPT-2: Sample

SYSTEM PROMPT (HUMAN-WRITTEN)	*Miley Cyrus was caught shoplifting from Abercrombie and Fitch on Hollywood Boulevard today.*
MODEL COMPLETION (MACHINE-WRITTEN, SECOND TRY)	The 19-year-old singer was caught on camera being escorted out of the store by security guards. The singer was wearing a black hoodie with the label 'Blurred Lines' on the front and 'Fashion Police' on the back. Scroll down for video Shoplifting: Miley Cyrus was caught shoplifting from Abercrombie and Fitch on Hollywood Boulevard today (pictured) The singer was wearing a black hoodie with the label 'Blurred Lines' on the front and 'Fashion Police' on the back The singer was also wearing a pair of black-rimmed glasses, a black jacket, black jeans and black sandals. She was carrying a pair of black and white striped gloves and a small black bag.

XLM: Multilingual Pre-Training

*Lample et al., 2019



What are we learning though?

- How can we understand what these models are doing?
  - What LM objective will help me for downstream task X?
- Look at the Neurons
- Probe
- Attention Weights
  - We will cover this in the tutorial!

Looking at Neuron Activations?

Cell that is sensitive to the depth of an expression:

```
#ifdef CONFIG_AUDITSYSCALL
static inline int audit_match_class_bits(int class, u32 *mask)
{
  int i;
  if (classes[class]) {
    for (i = 0; i < AUDIT_BITMASK_SIZE; i++)
      if (mask[i] & classes[class][i])
        return 0;
  }
  return 1;
}
```

*Karpathy et al., 2016

Probing

- Fix our contextual representations and train a single layer on a downstream task
- YMMV: Does not perform well on NER, grammatical error detection, and conjunct identification (Liu et al., 2019)

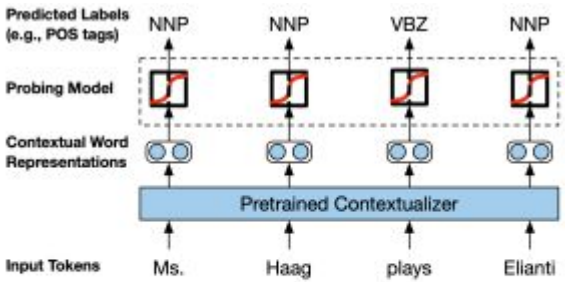


Figure 1: An illustration of the probing model setup used to study the linguistic knowledge within contextual word representations.

*Liu et al., 2019

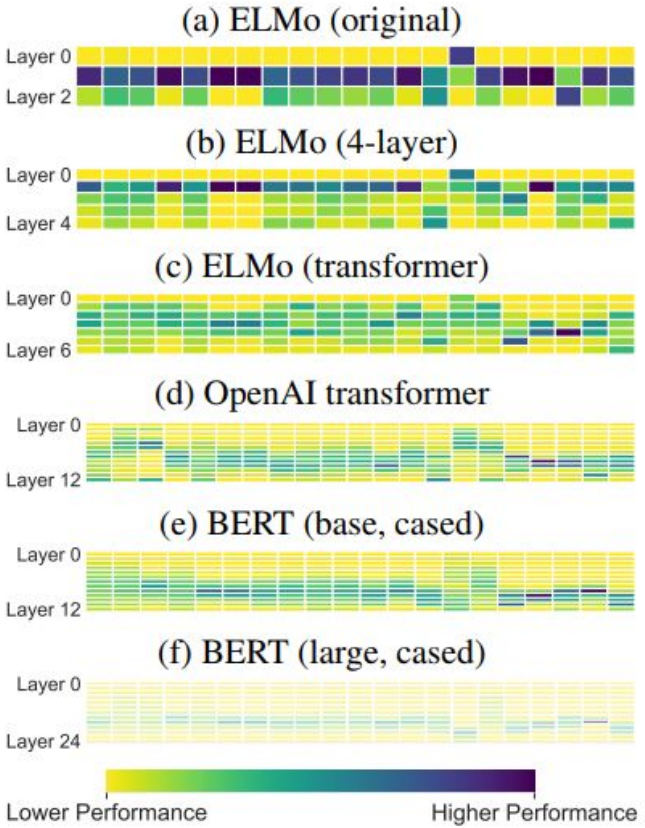
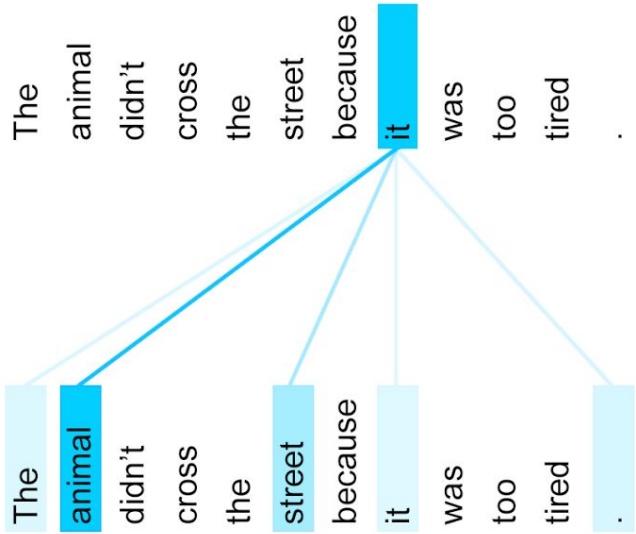
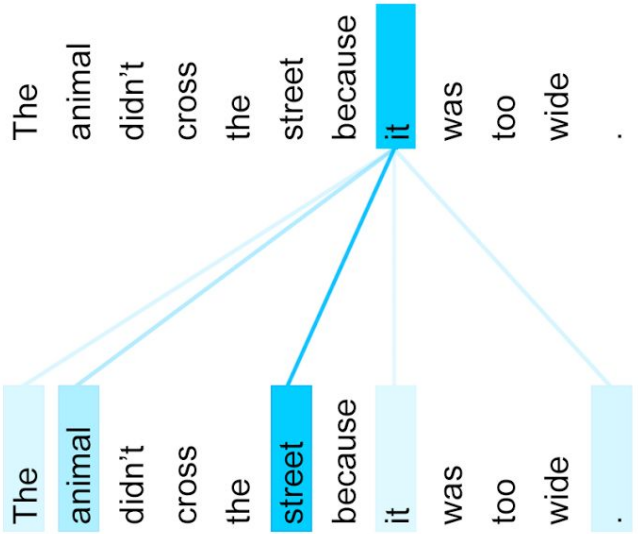


Figure 3: A visualization of layerwise patterns in task performance. Each column represents a probing task, and each row represents a contextualizer layer.

Attention Weight Visualization

- For Transformers, we can take a look at the attention heads individually
  - These weights do inform us of what the model is learning!





Catastrophic Forgetting

- Problem: as we are learning in-domain (downstream) task, we are “forgetting” what we learned in the general purpose one
  - Most applications may not care
  - If we do care, use multi-task learning to prevent overfitting
    - How to select?

Moving Forward

- Exciting time in NLP!
- Generative Pre-Training and Transfer Learning are powerful tools that have transformed NLP
  - They require a significant amount of time to train
  - The pre-training objectives are important to downstream tasks
- Transformer-type architectures are trending up
  - LSTM-based models like ELMo are less popular
  - Transformers are computationally efficient to train
  - Slightly easier to interpret than LSTMs
- Scratching surface of downstream performance but…
  - these models already have significant power as shown in benchmarks like GLUE

References

- Announcing SyntaxNet: The World's Most Accurate Parser Goes Open Source
  - Petrov, 2016
- Left-to-Right Dependency Parsing with Pointer Networks
  - Fernandez-Gonzalez and Gomez-Rodriguez, 2019
- Stack-Pointer Networks for Dependency Parsing
  - Ma et al., 2018
- Deep Biaffine Attention for Neural Dependency Parsing
  - Dozat & Manning, 2016
- Simple and Accurate Dependency Parsing Using Bidirectional LSTM Feature Representations
  - Kipperwasser & Goldberg, TACL, 2016
- Visualizing and Understanding Recurrent Neural Networks
  - Karpathy et al., 2015
- A Fast and Accurate Dependency Parser using Neural Networks
  - Chen & Manning, EMNLP, 2014

References

- Named Entity Recognition through Classifier Combination
  - Florian et al., 2003
- A Framework for Learning Predictive Structures from Multiple Tasks and Unlabeled Data
  - Ando & Zhang, 2005
- Phrase Clustering for Discriminative Learning
  - Lin & Wu, 2009
- Natural Language Processing (almost) from Scratch
  - Collobert et al., 2011
- Lexicon Infused Phrase Embeddings for Named Entity Resolution
  - Passos et al., 2014
- Semi-supervised sequence tagging with bidirectional language models
  - Peters et al., 2017
- Named Entity Recognition with Bidirectional LSTM-CNNs
  - Chiu & Nichols, 2015
- End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF
  - Ma & Hovy, 2015

References

- Neural Architectures for Named Entity Recognition
  - Lample et al., 2016
- Multi-Task Cross-Lingual Sequence Tagging from Scratch
  - Yang et al., 2017
- Attention Is All You Need
  - Vaswani et al., 2017
- Deep Contextualized Word Representations
  - Peters et al., 2018
- BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding
  - Devlin et al., 2019
- Improving Language Understanding by Generative Pre-Training
  - Radford et al., 2017
- Language Models are Unsupervised Multitask Learners
  - Radford et al., 2019
- Cross-lingual Language Model Pretraining
  - Lample & Conneau, 2019

References

- To Tune or Not to Tune? Adapting Pretrained Representations to Diverse Tasks
  - Peters, Ruder, Smith, 2019
- Cloze-driven Pretraining of Self-attention Networks
  - Baevski et al., 2019
- NAACL2019 Transfer Learning Tutorial
  - Ruder, Peters, Swayamdipta, Wolf, NAACL, 2019
- Neural Machine Translation by Jointly Learning to Align and Translate
  - Bahdanau, Cho & Bengio, 2014
- Sequence to Sequence Learning with Neural Networks
  - Sutskever, Vinyals & Le, 2014
- Linguistic Knowledge and Transferability of Contextual Representation
  - Liu et al., 2019
- https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270