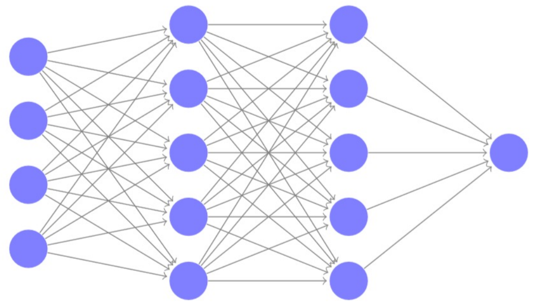
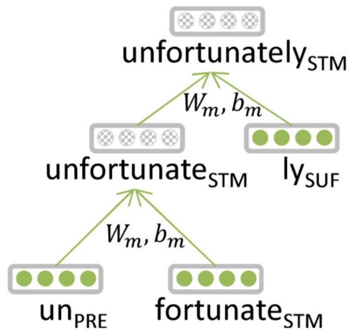
# Seq2Seq Models & Transfer Learning

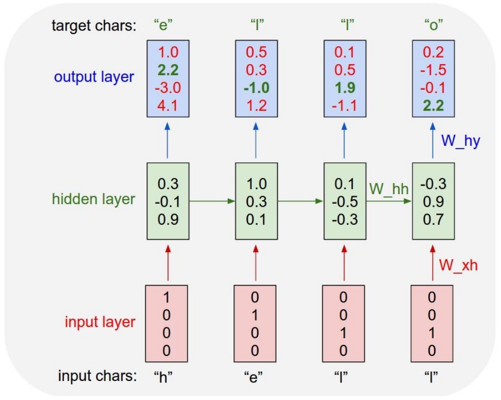
AUSTIN BLODGETT

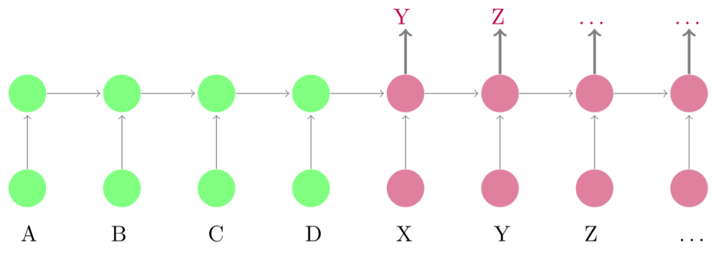
## Brief Review

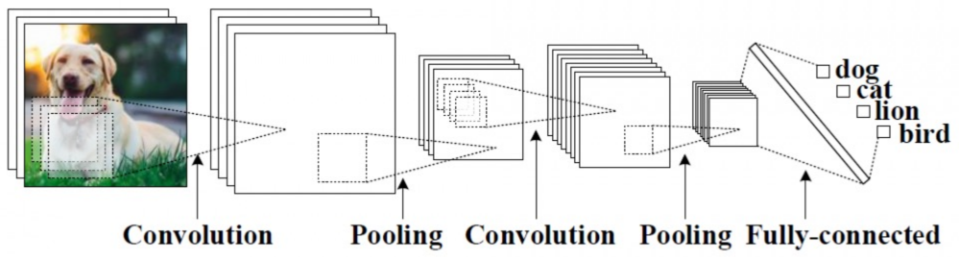
# Neural Algorithms











| NN Task | Example Input | Example Output |
|---|---|---|
| Binary Classification | features | +/- |
| Multiclass Classification | features | decl, imper, inter, … |
| Sequence Classification | sentence | POS tags |
| Sequence to Sequence | (English) sentence | (Spanish) sentence |
| Structured Prediction | sentence | dependency tree or AMR parse |

| NN Task | Example Input | Example Output |
|---|---|---|
| Binary classification | features | +/- |
| Multiclass classification | features | decl, imper, inter, … |
| Sequence | sentence | POS tags |
| Sequence to Sequence | (English) sentence | (Spanish) sentence |
| Tree/Graph Parsing | sentence | dependency tree or AMR parse |

# Seq2Seq Tasks

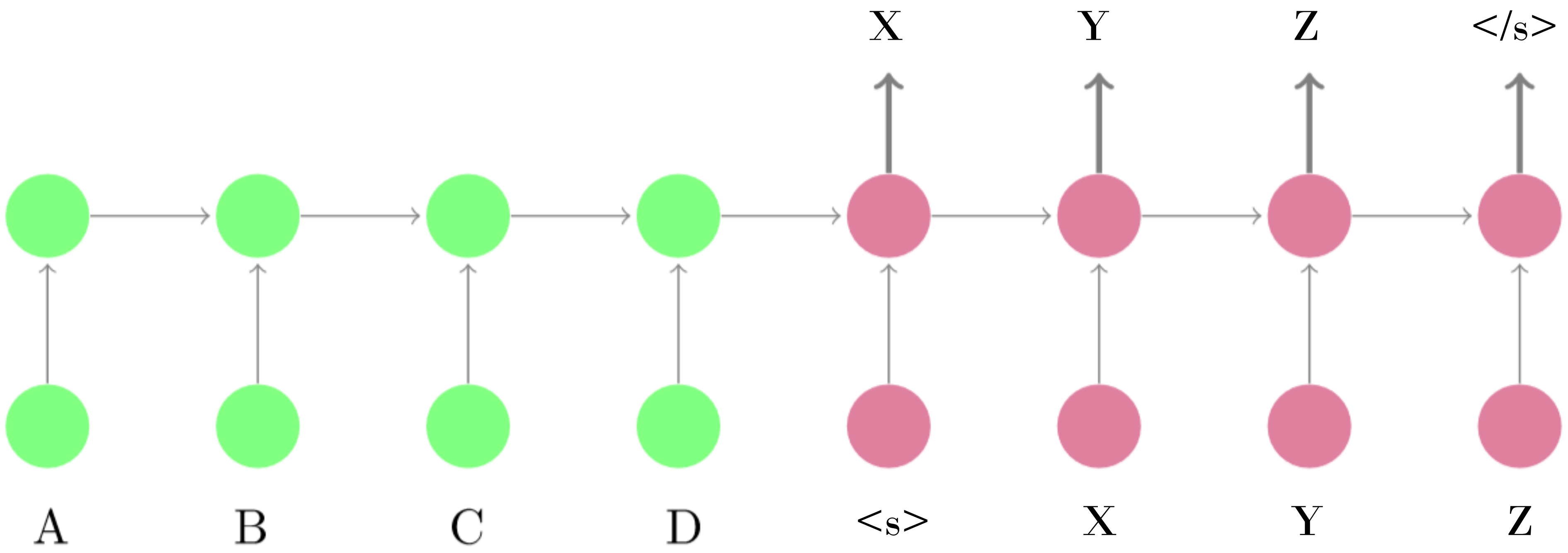
- Tasks
  - Machine Translation
  - Automatic Dialogue
  - Question Answering
  - Document Summarization
  - (Some) Semantic Parsing

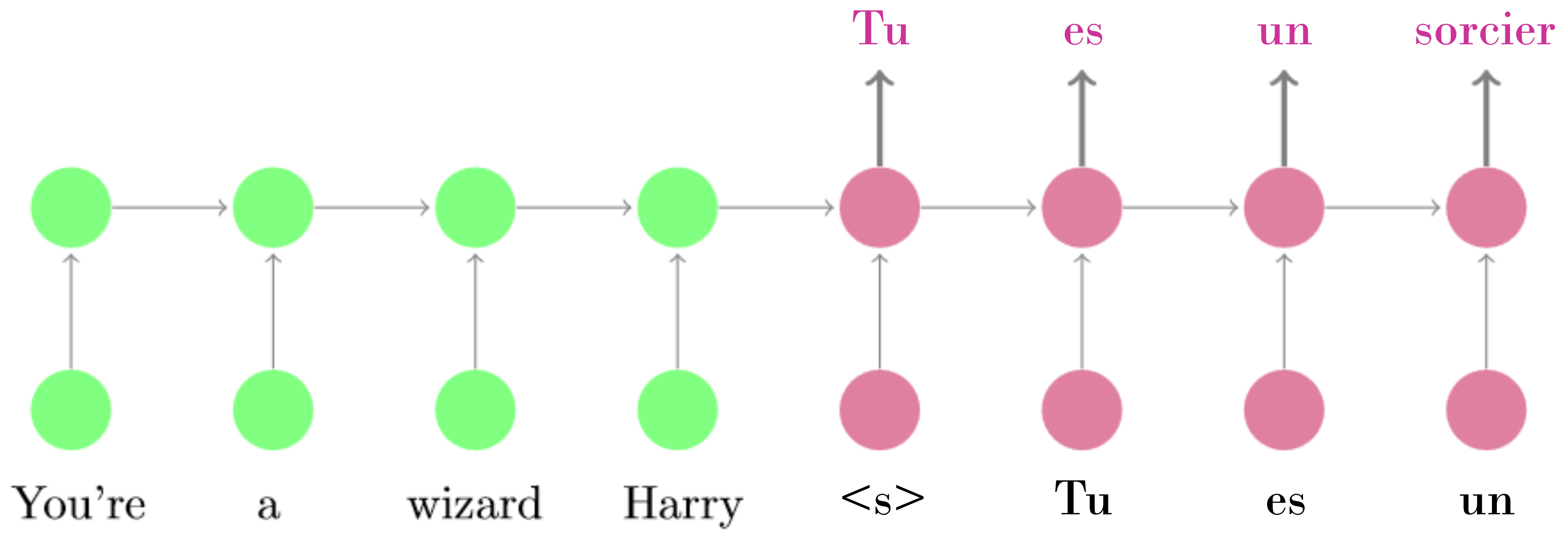
## Encoder-Decoder models

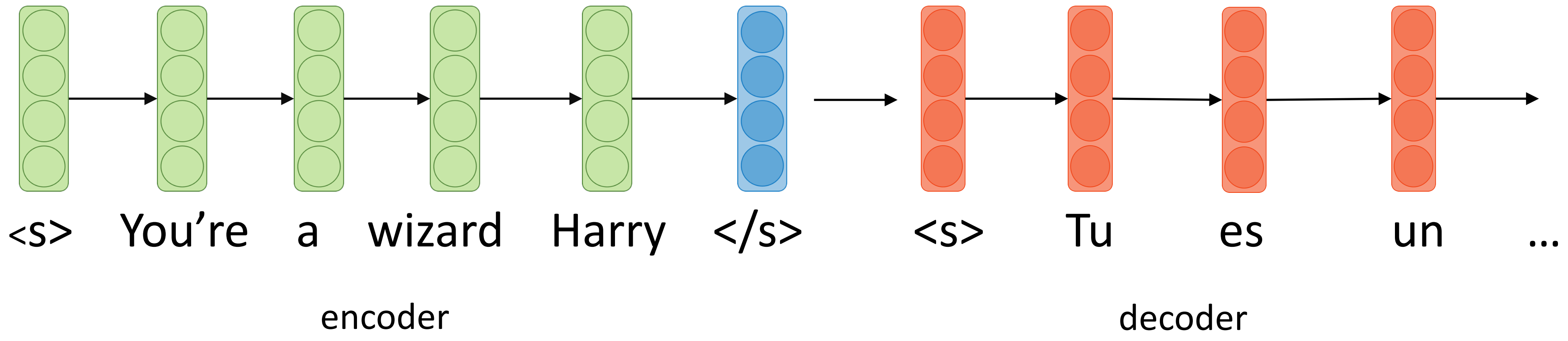
- **Encoder-Decoder model** (also **Seq2Seq**) – Take a sequence as input and predict a sequence as output
- *Input and Output may be different lengths*
- Encoder (*RNN*) models input, Decoder (*RNN*) models output

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). **Sequence to sequence learning with neural networks**. In *Advances in neural information processing system.*

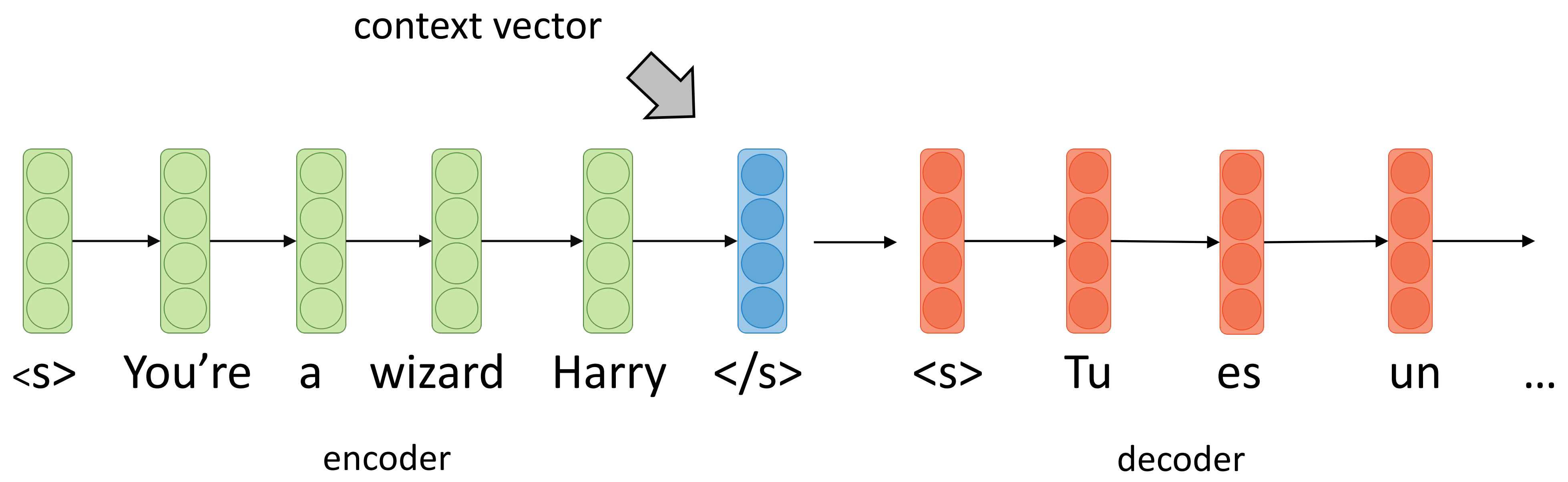
Cho, K., et al. (2014). **Learning phrase representations using RNN encoder-decoder for statistical machine translation**.


X
Y
Z
</s>
A
B
C
D
<s>
X
Y
Z

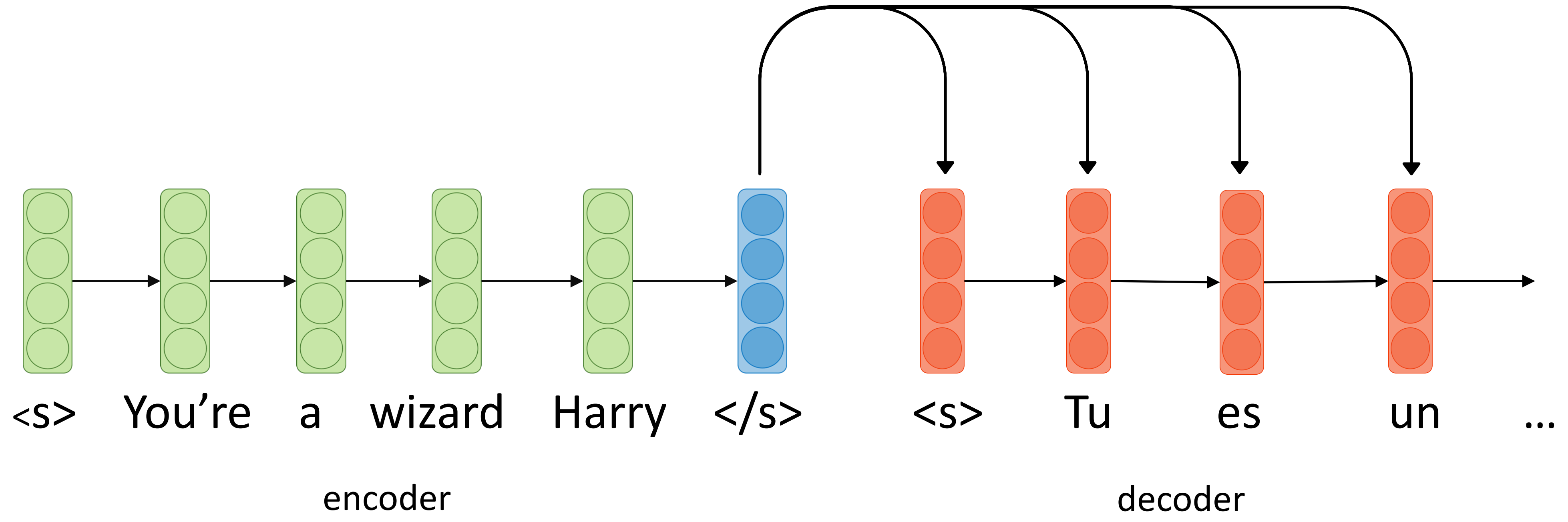

Tu
es
un
sorcier
You're
a
wizard
Harry
<s>
Tu
es
un



<s>
You’re
a
wizard
Harry
</s>
<s>
Tu
es
un
…
encoder
decoder


context vector
<s>
You're
a
wizard
Harry
</s>
<s>
Tu
es
un
...
encoder
decoder

Cho et al., (2014): connect to every state in decoder



## Seq2Seq with RNNs

- RNNs are slow: each timestep needs the first to be completed
- RNNs struggle to capture long-distance dependencies:
  - The **keys** to the door **are**/is gray
  - Caused by issues in managing hidden state: when is it OK to forget that “keys” is the subject and is plural?

## Attention

- A way of allowing tokens to “explicitly” represent their dependent words
  - Intuition: a continuous version of word alignments
- For a given word in the output, represents the importance of each word in the input
- We’ll talk about how much the network “attends” to each word.
- First used in MT, improved BLEU score by 10 pts (huge!)

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. (2014) **Neural machine translation by jointly learning to align and translate**.

## Activity

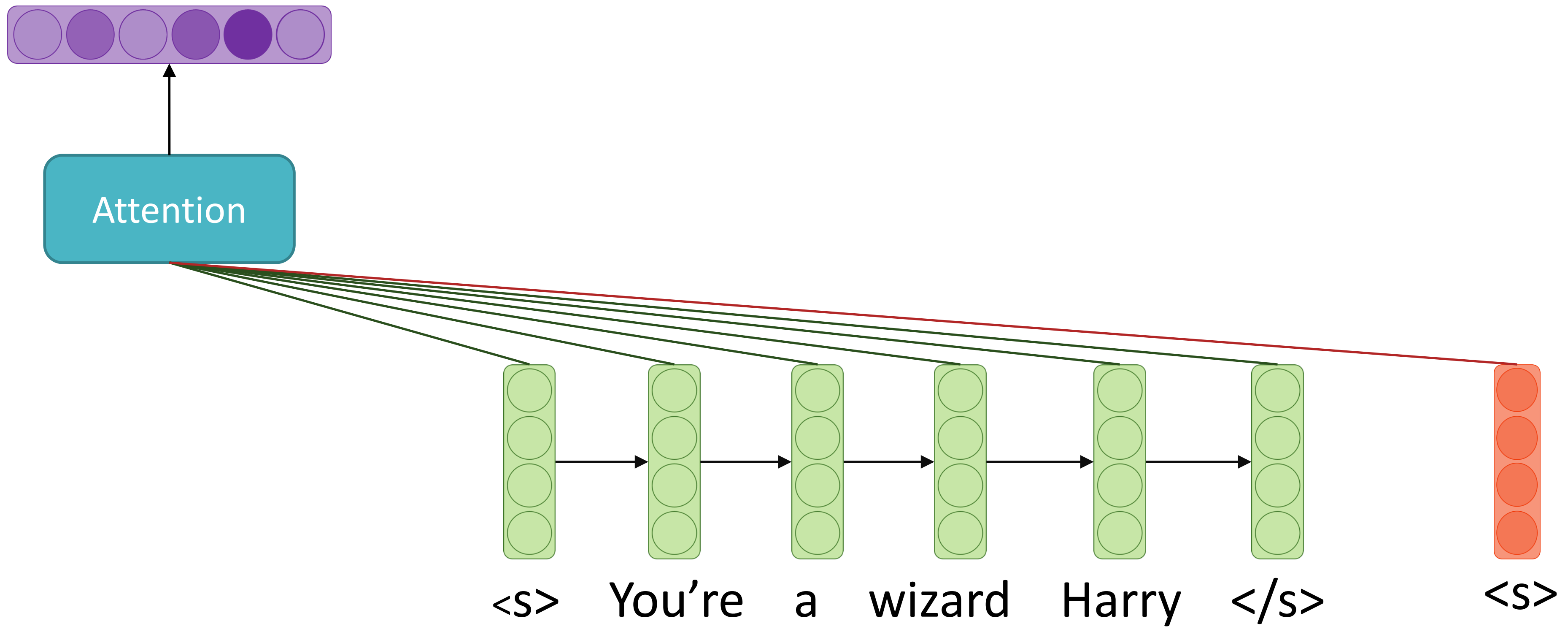
Question

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| What | is | the | preferred | weapon | of | the | Jedi | ? |
| PRON | AUX | DET | ADJ | NOUN | ADP | DET | PROPN | |

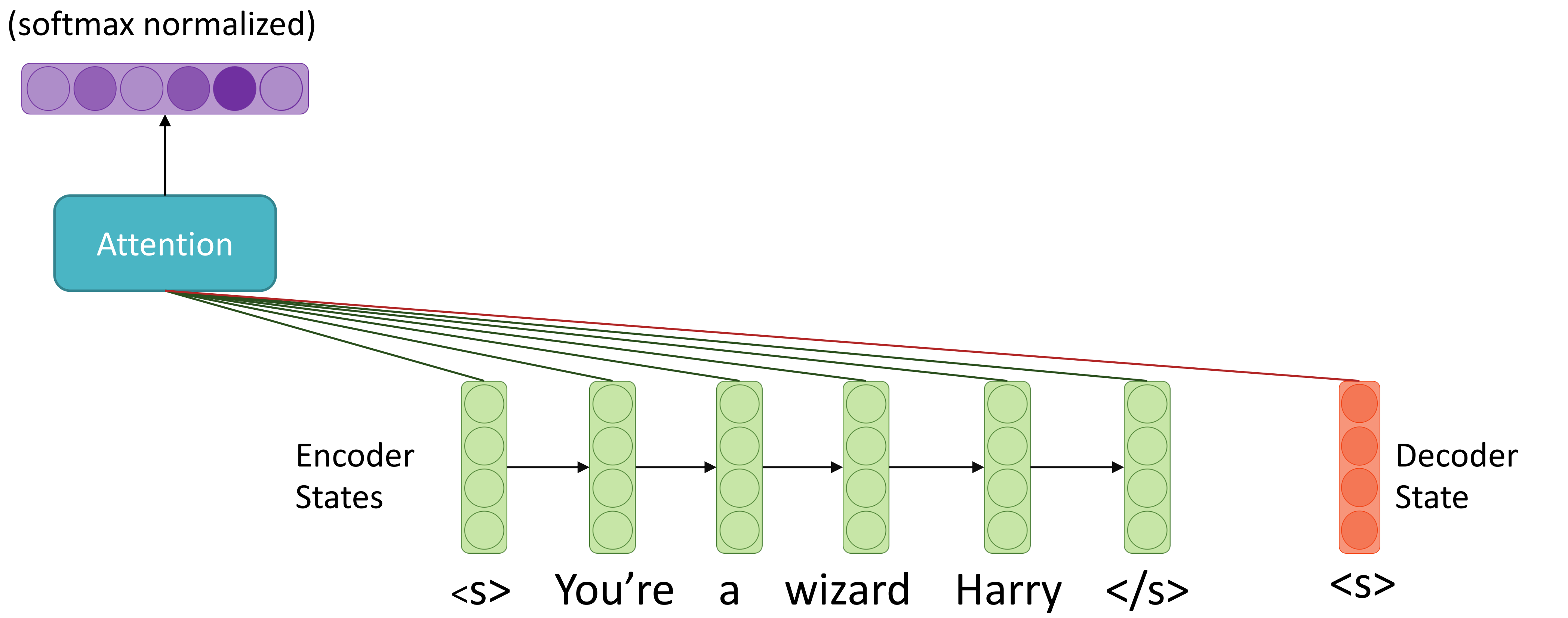
Answer

| | | |
|---|---|---|
| a | light | saber |
| 1 | 2 | 3 |

# Attention



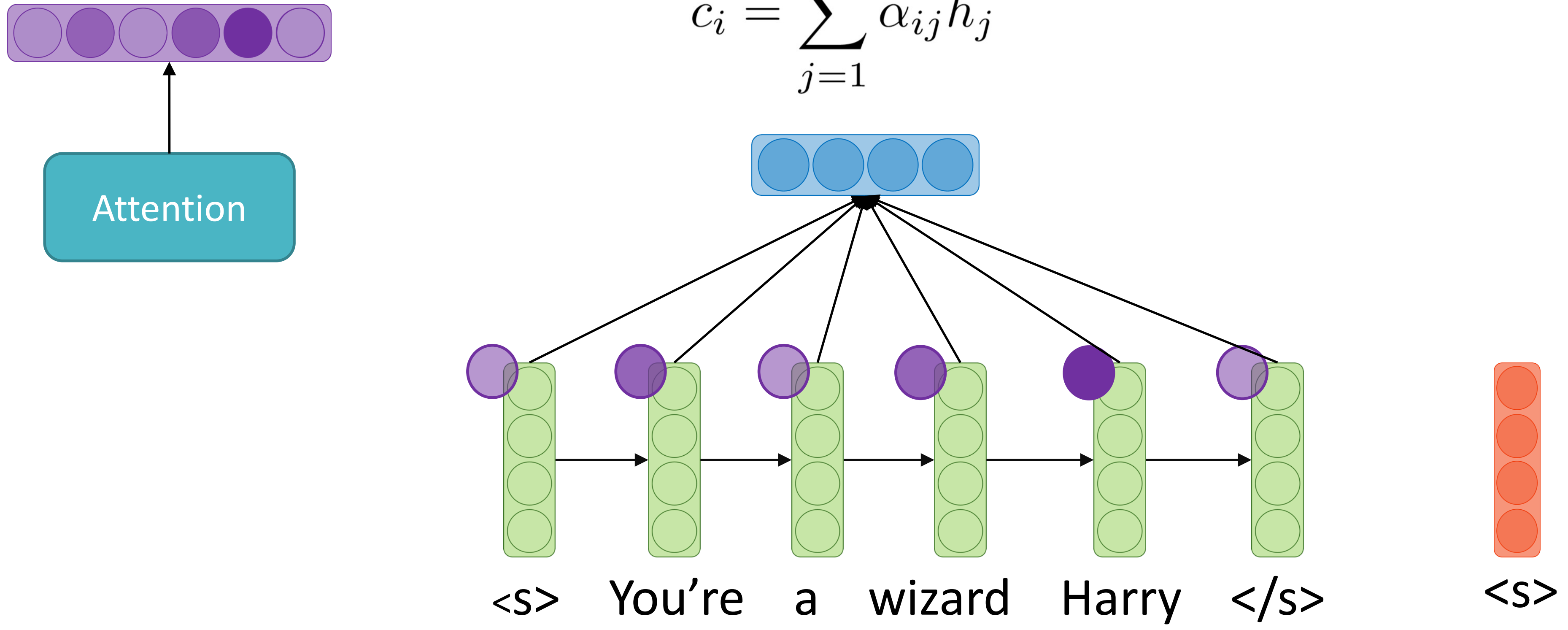
# Attention


(softmax normalized)
Attention
Encoder States
Decoder State
<s>
You’re
a
wizard
Harry
</s>
<s>

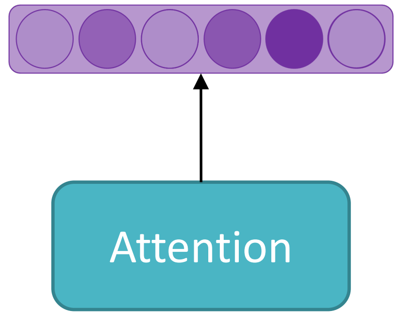
# Attention

Context vector

$$c_i = \sum_{j=1} \alpha_{ij} h_j$$

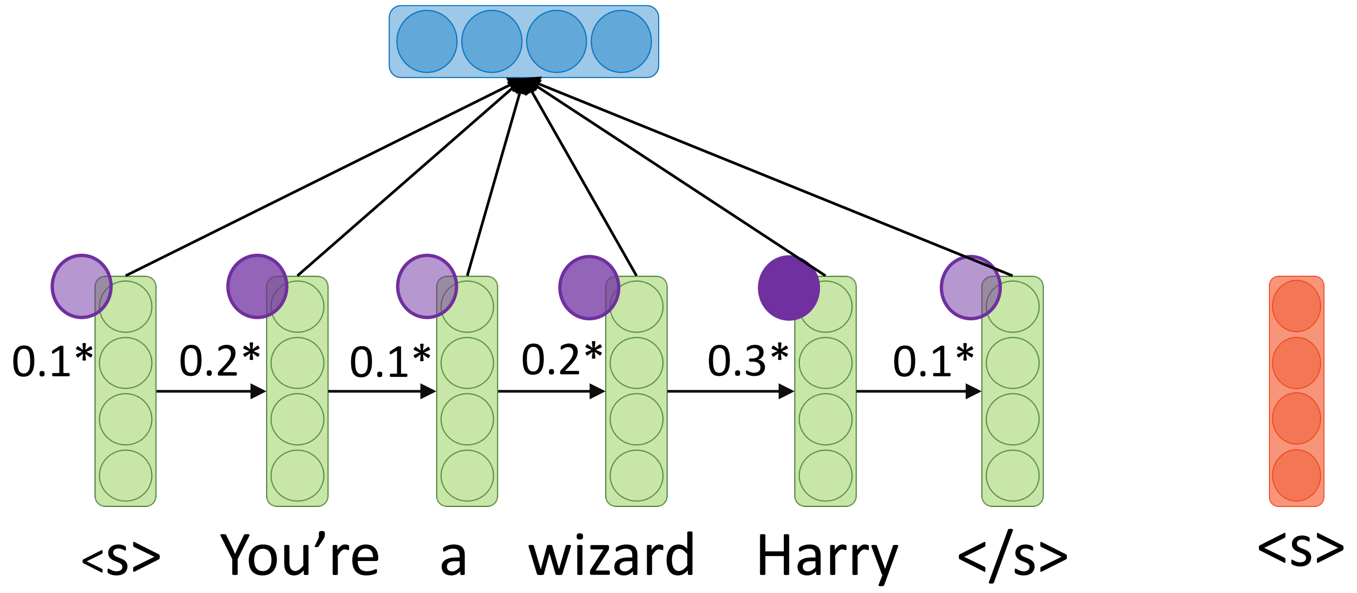


# Attention



Context vector

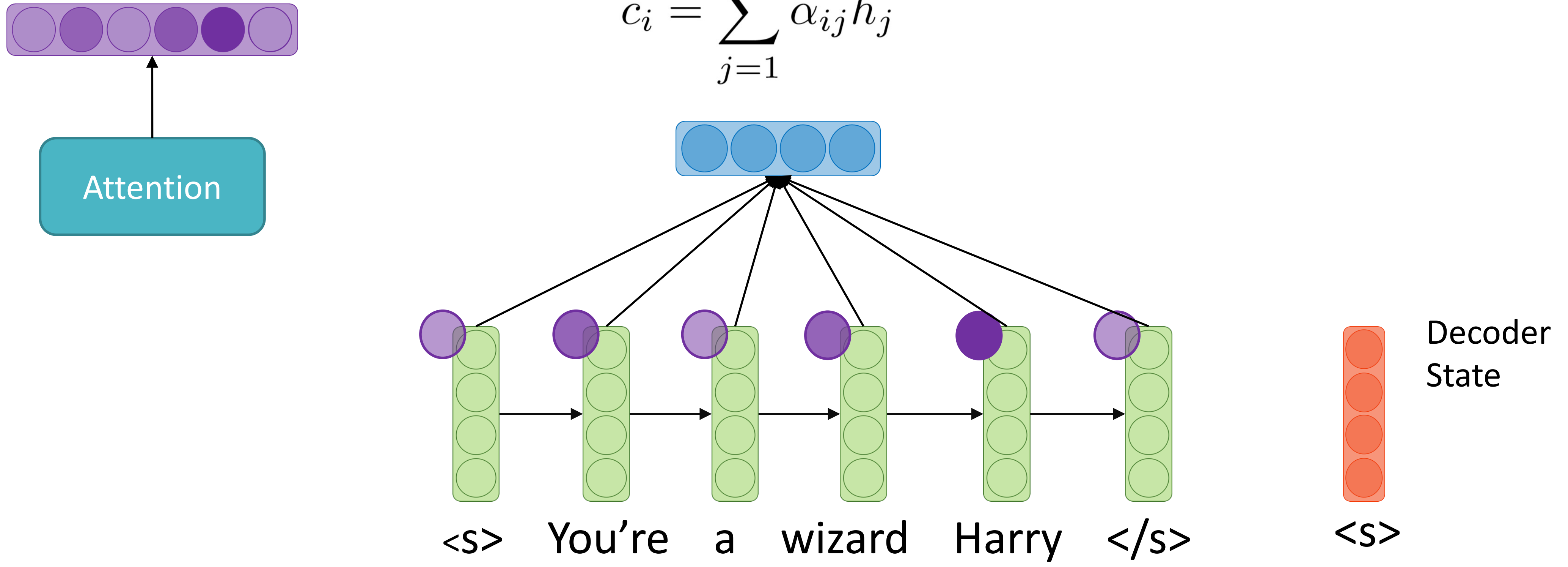
$$c_i = \sum_{j=1} \alpha_{ij} h_j$$



# Attention

Context vector

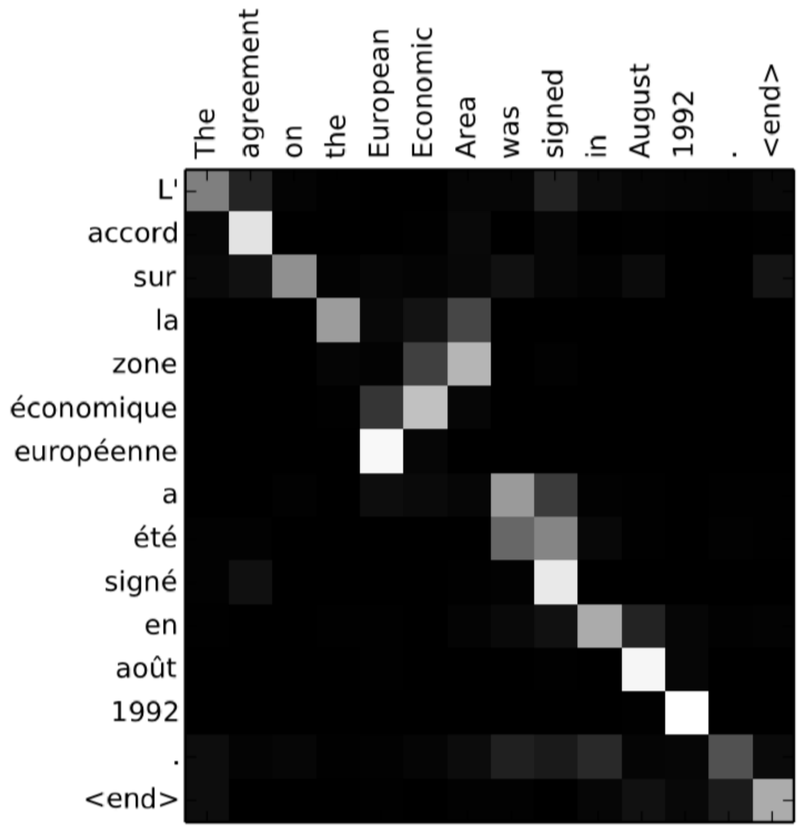
$$c_i = \sum_{j=1} \alpha_{ij} h_j$$



# Attention

Every attention value depends on one word in the source and one in the target.

Attention matrix tells us how “important” a source word is for each target word (much like alignment).



## Attention vs. RNN

- Shift “context management” into its own module (attention)
- Allow decoder state to handle everything else (e.g. grammar of the target language)
- Benefits
  - More parallelizable
  - No more “remembering” problem: each token gets its own context vector that is more independent of other tokens’ context vectors

## Transformers

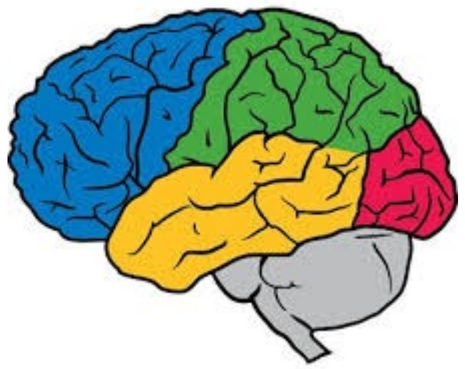
# Attention is All You Need (Vaswani et al., 2017)

Debut of the **Transformer** architecture

The same model used in:

- BERT (Devlin, et al. 2018)
- LISA (Strubell, et al. 2018)
- RoBERTa (Liu et al., 2019)
- and others…

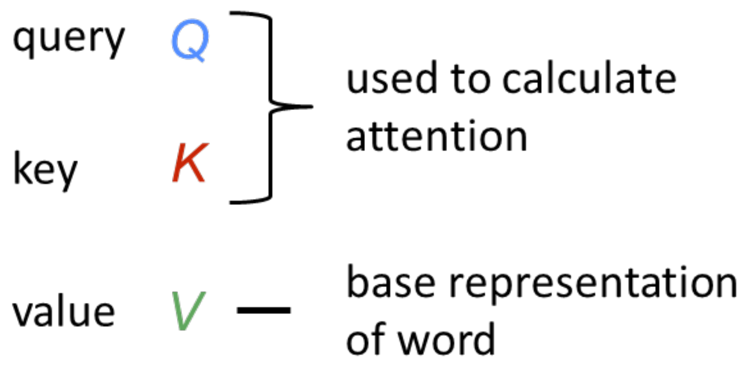
Motto paraphrased: *No more RNNs, CNNs, just use Attention!*



## Introducing Self-Attention

- Shift “context management” into its own module (attention)
- Allow decoder state to handle everything else (e.g. grammar of the target language)
- Benefits
  - More parallelizable
  - No more “remembering” problem: each token gets its own context vector that is more independent of other tokens’ context vectors

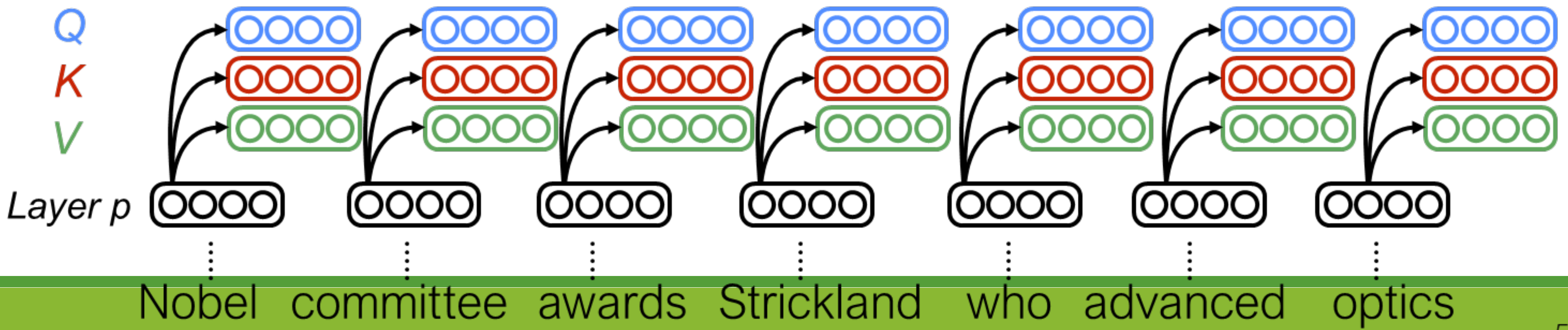
# Transformer: Self-Attention



Slide from Strubell, et al. (2018).

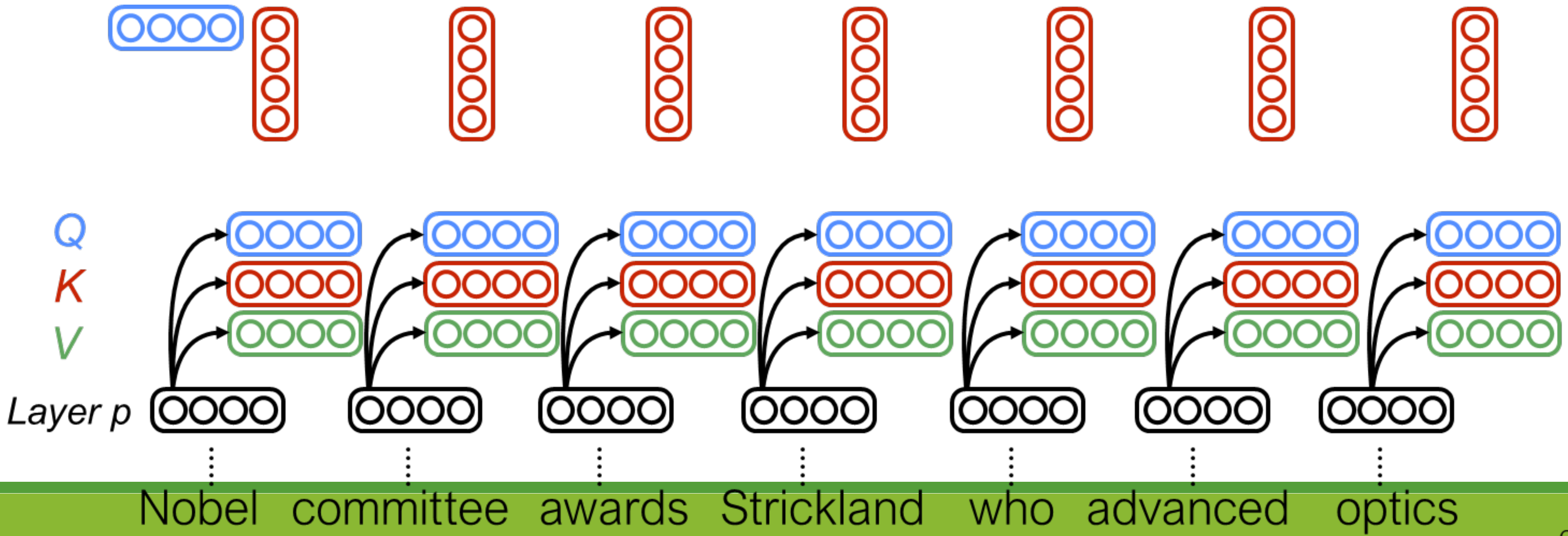
# Transformer: Self-Attention

(**Q**, **K**, **V** are trainable parameters)



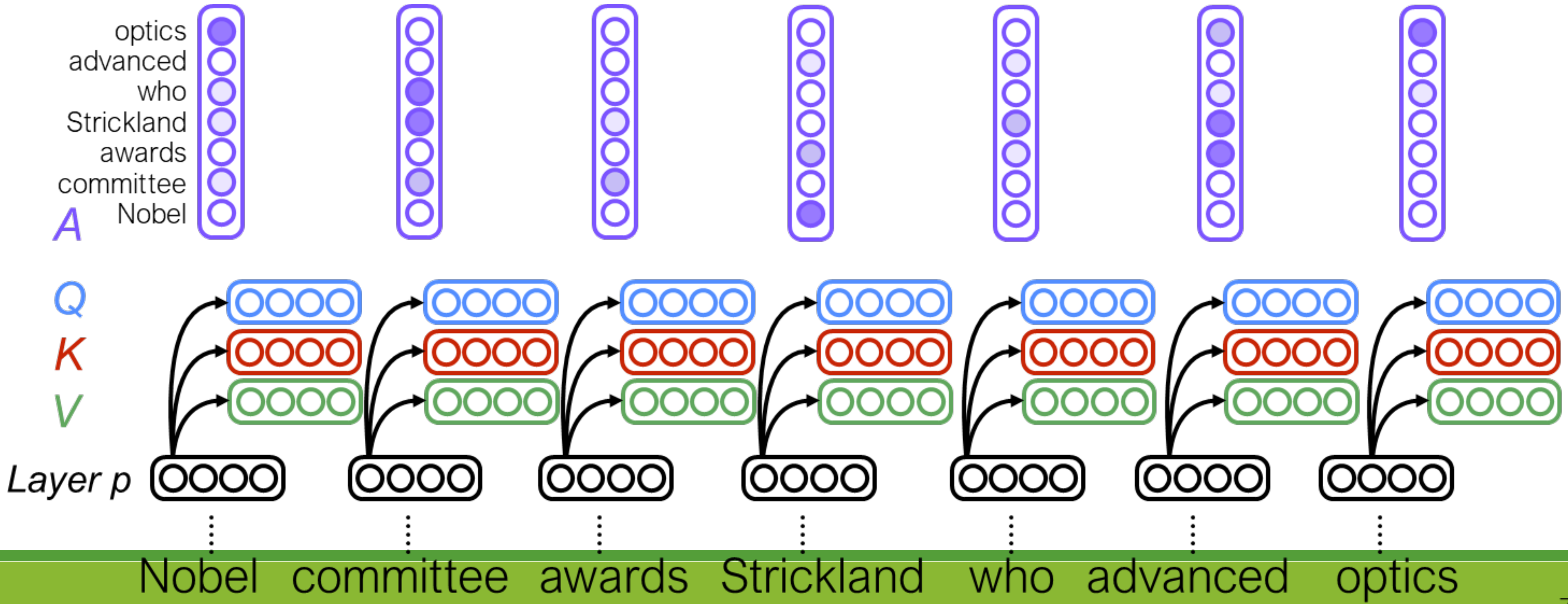
Slide from Strubell, et al. (2018).

# Transformer: Self-Attention



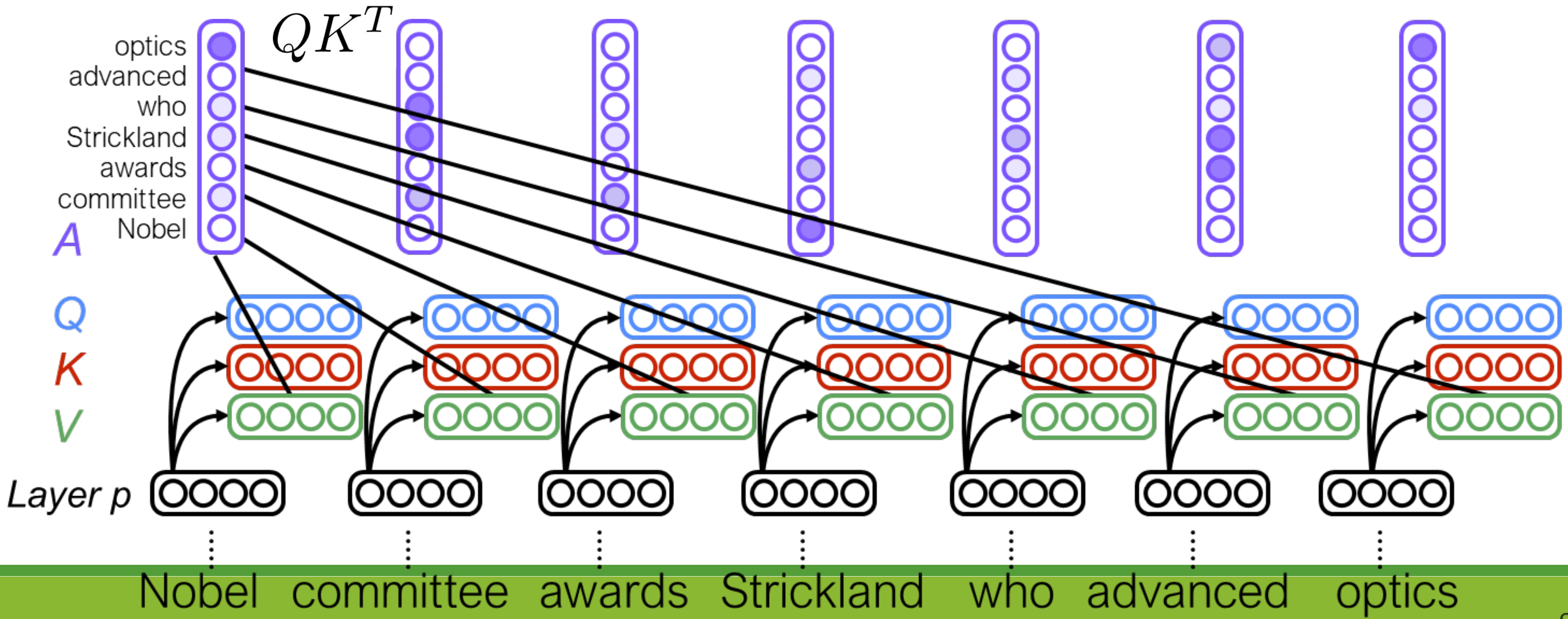
Slide from Strubell, et al. (2018).

# Transformer: Self-Attention



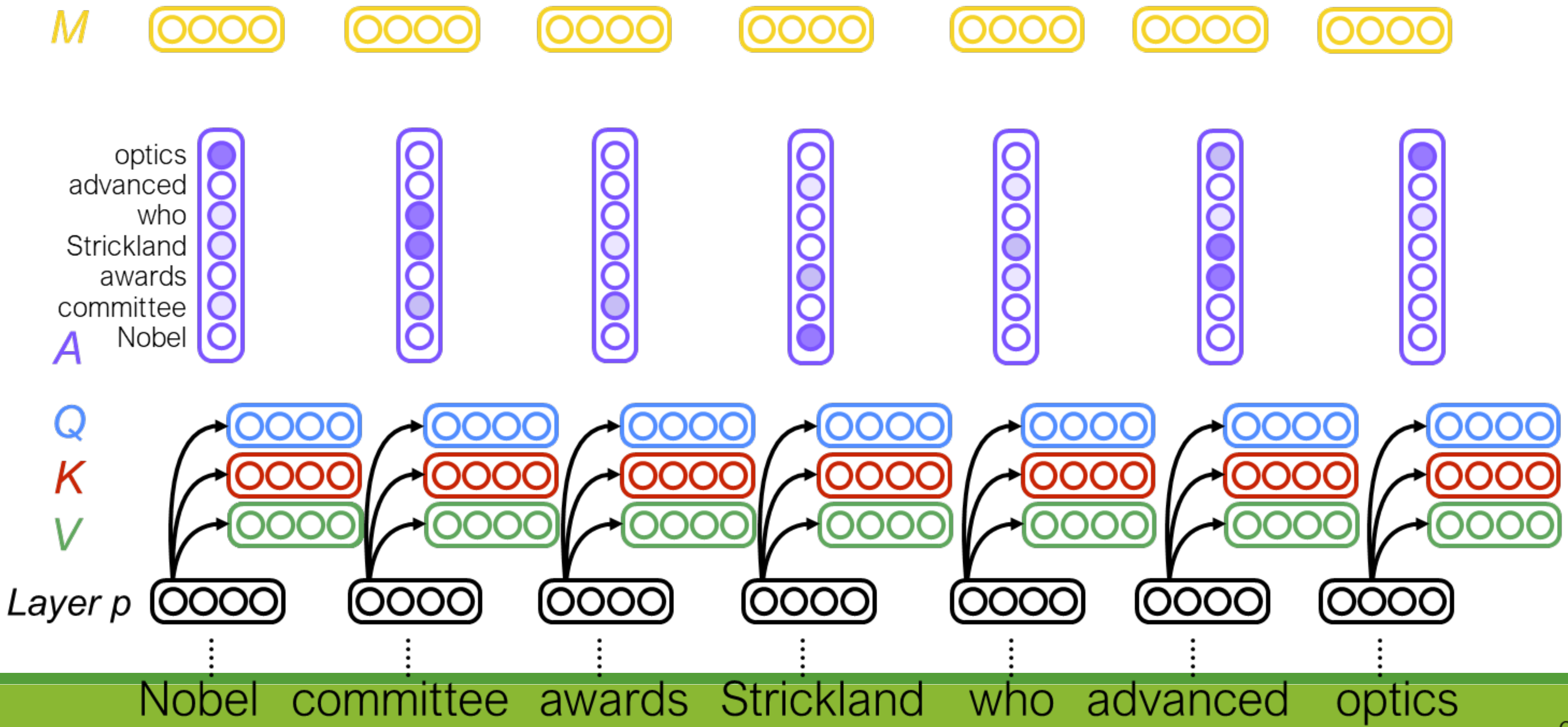
Slide from Strubell, et al. (2018).

# Transformer: Self-Attention



Slide from Strubell, et al. (2018).

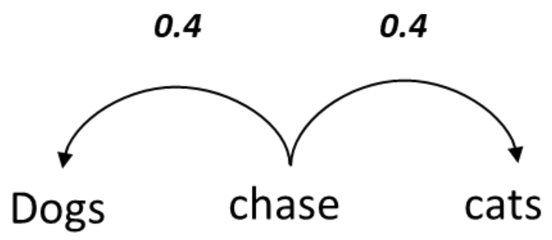
## Transformer: Self-Attention



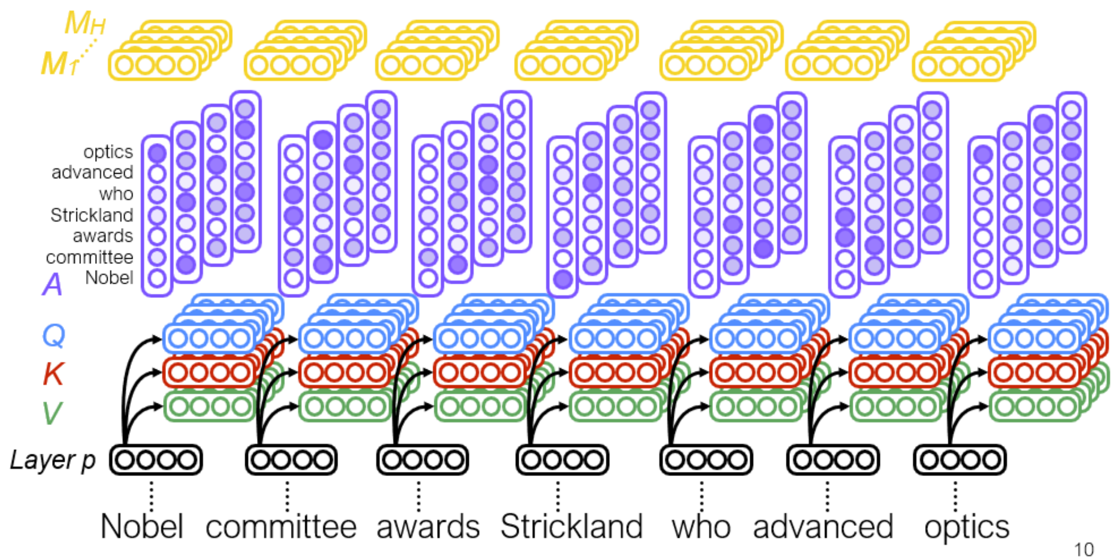
# Multi-Head Attention

How to distinguish dependencies:

One attention layer can't distinguish two dependencies (subject vs. object).



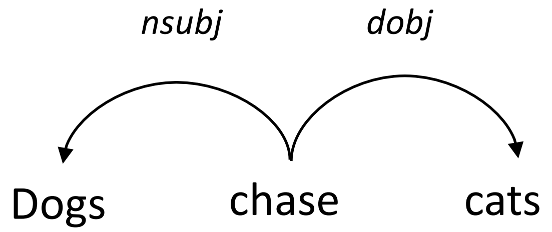
Use multiple attention layers, hopefully one represents subject, one object, etc.

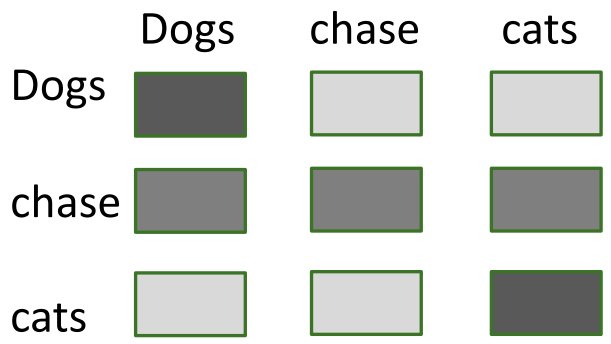


## Linguistically Motivated

### Strengths

- Captures long-distance dependencies!
- Intuitively: approximates *weighted unlabeled dependencies*

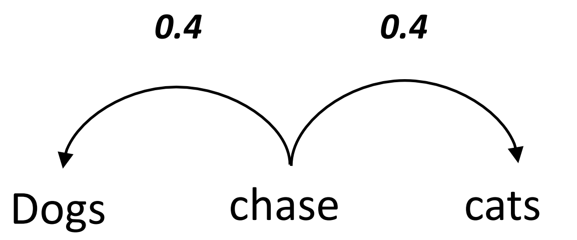


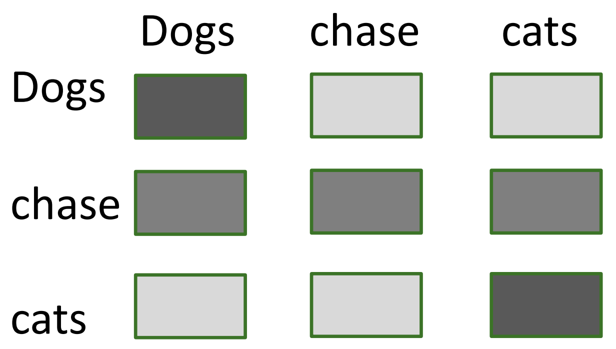


# Linguistically Motivated

## Strengths

- Captures long-distance dependencies!
- Intuitively: approximates *weighted unlabeled dependencies*

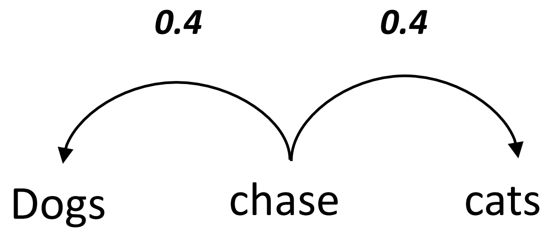




## Linguistically Motivated

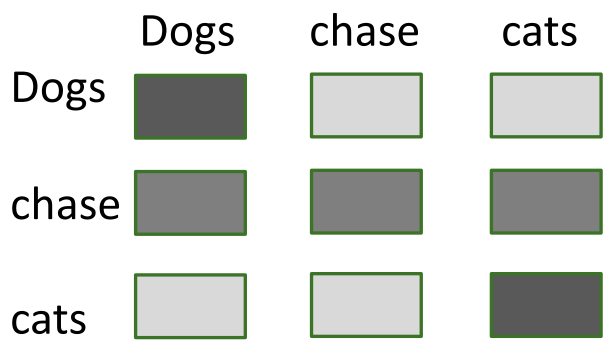
### Strengths

- Captures long-distance dependencies!
- Intuitively: approximates *weighted unlabeled dependencies*



### Weaknesses (addressed in next slides)

- Weak model of word order
- One layer can't distinguish dependencies
- No locality bias



## Transformer Architecture

- Encode-Decoder with Transformers instead of RNNs
- Large improvement over LSTM encoder-decoder. Why?
  - long-distance relations
  - better representation of syntax
  - faster to train (when using TPUs)

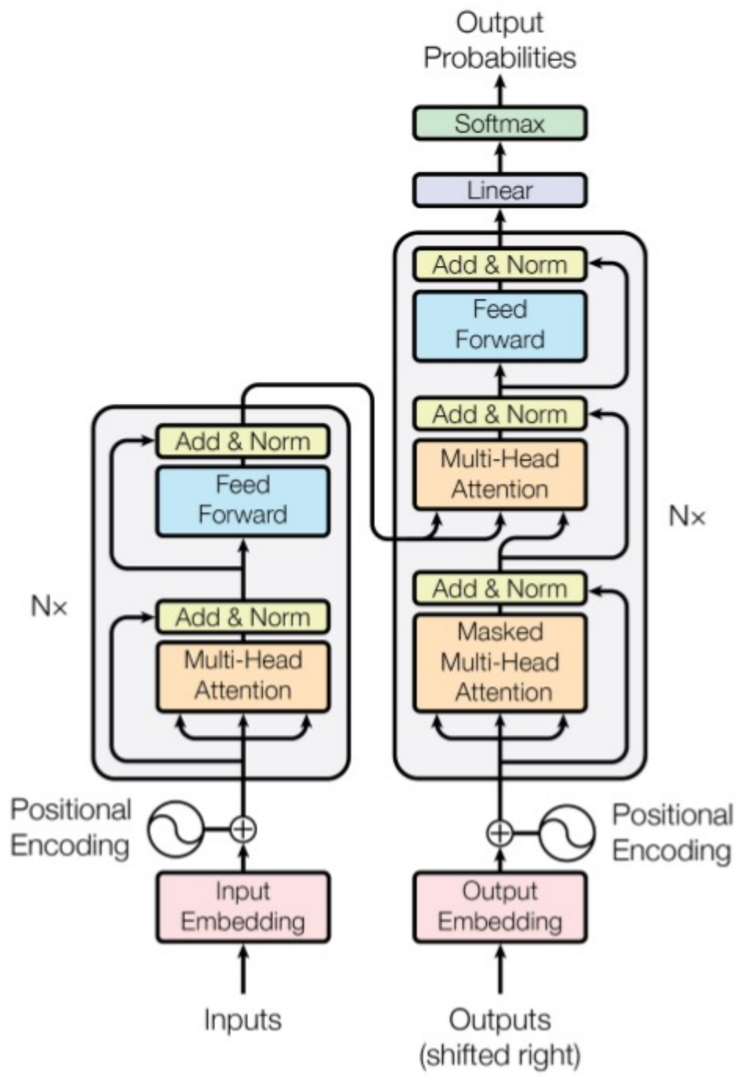


Figure 1: The Transformer - model architecture.

## Positional Encoding

Caveat: self-attention on its own doesn’t have awareness of word order, needs to be represented another way (position embeddings)

## Replicate, Extend, etc.

Tensorflow

https://github.com/tensorflow/tensor2tensor

Pytorch

https://github.com/jadore801120/attention-is-all-you-need-pytorch

Annotated Code

http://nlp.seas.harvard.edu/2018/04/03/attention.html

Illustrated Explanation

http://jalammar.github.io/illustrated-transformer/

## Transfer Learning: ELMo & BERT

# Learning Paradigms so far

- rule-based (symbolic) approaches
- supervised learning
- **self-**supervised learning
- (unsupervised learning)

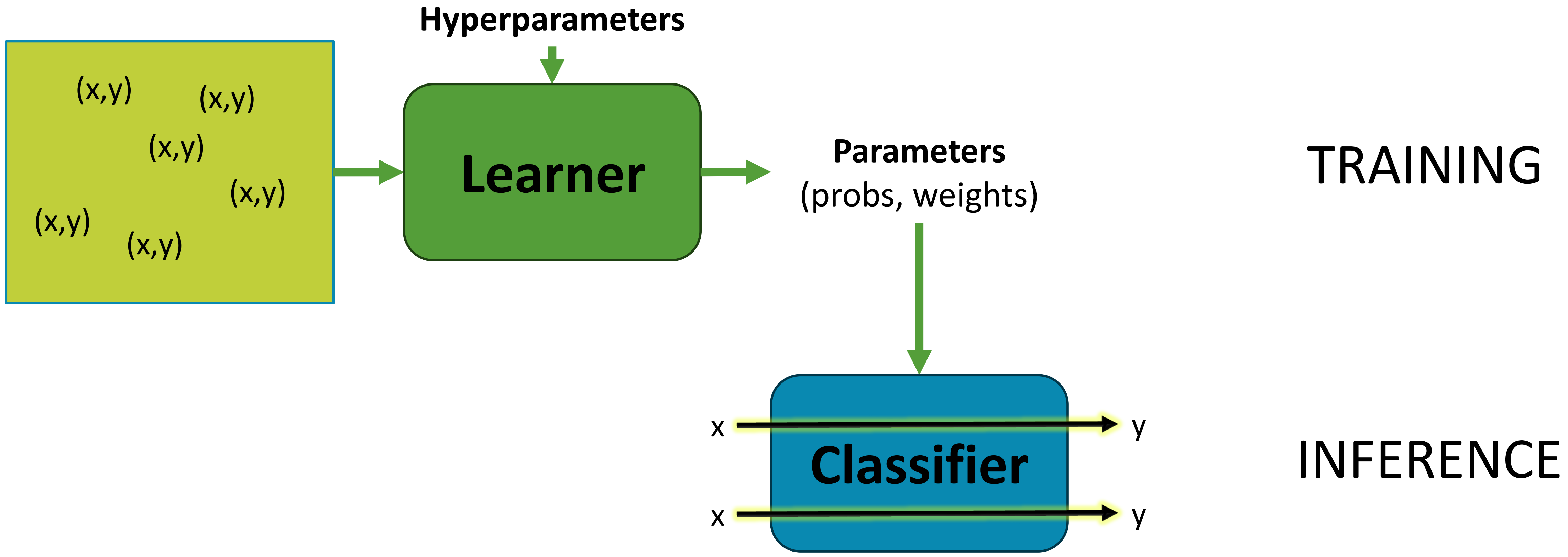
*Categorization exercise:*

LSTM POS tagger; Naïve Bayes sentiment analyzer; Regex tokenizer;
N-gram LM; word2vec embeddings; CFG syntactic parser

## Learning Paradigms so far

- rule-based (symbolic) approaches
  - Deterministic procedure e.g. regular expression to match a pattern, or parsing with a plain CFG
  - Good e.g. for format conversion; English tokenization
- supervised learning
  - Labeled training data
  - E.g. linear classifiers; prob. models like HMM/PCFG; RNN for classification/tagging
- **self-**supervised learning, representation learning
  - Goal is not labeling, but predicting/encoding/scoring text
  - static word embeddings, language models
- (unsupervised learning: clustering, topic models)

# Supervised Learning (inductive)



## Review: Word Embeddings

**GloVe** is a collection of pretrained (static) word embeddings that can be plugged into your models.

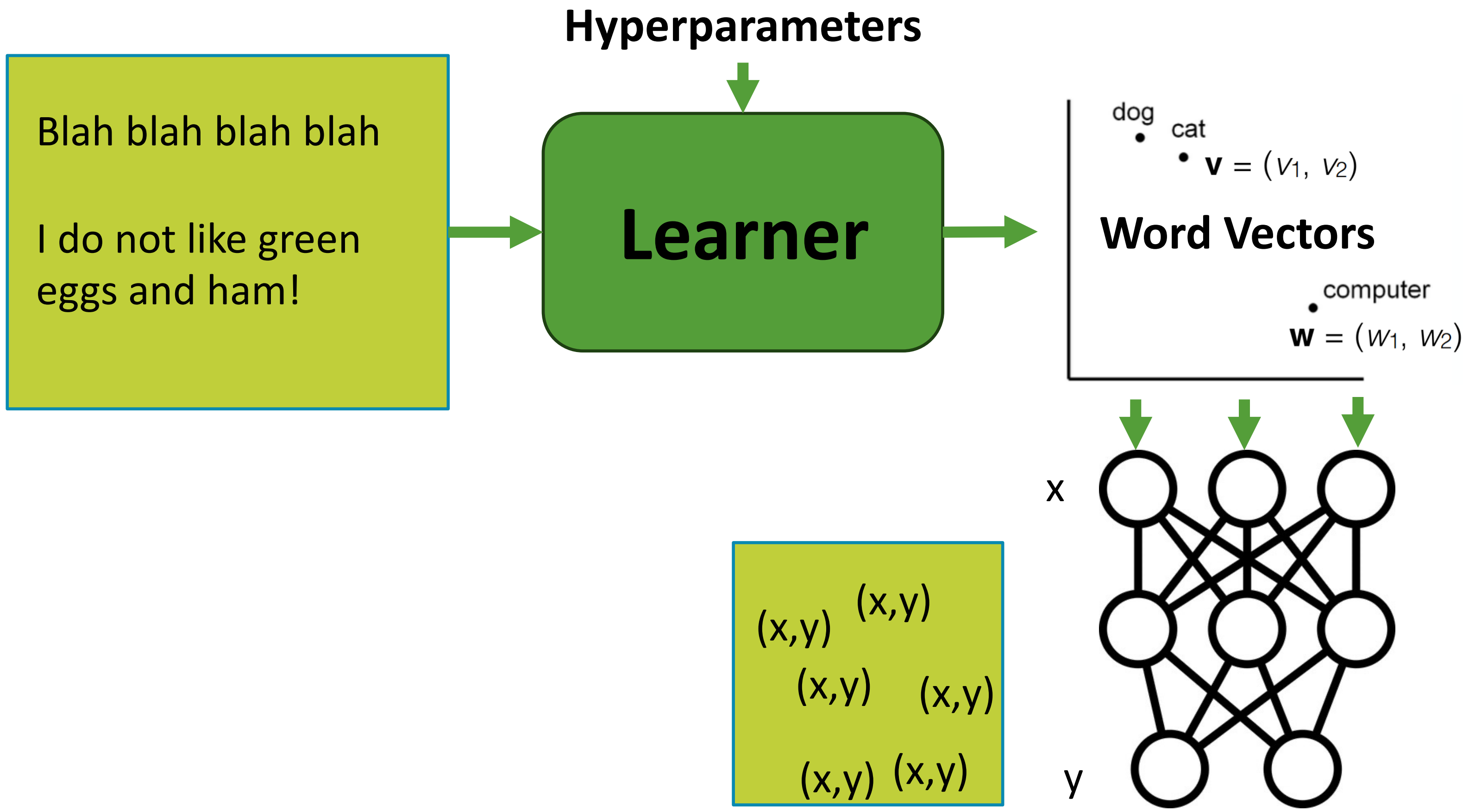
Approximate semantic features: King - Man + Woman = Queen

Trained on millions of sentences

Can be “tuned”: your model can adjust GloVe features to be more useful for your task.

Pennington, J., Socher, R., & Manning, C. (2014). **Glove: Global vectors for word representation**. In *Proceedings of the 2014 conference on EMNLP.*

# Deep Representation Learning (e.g. word2vec)



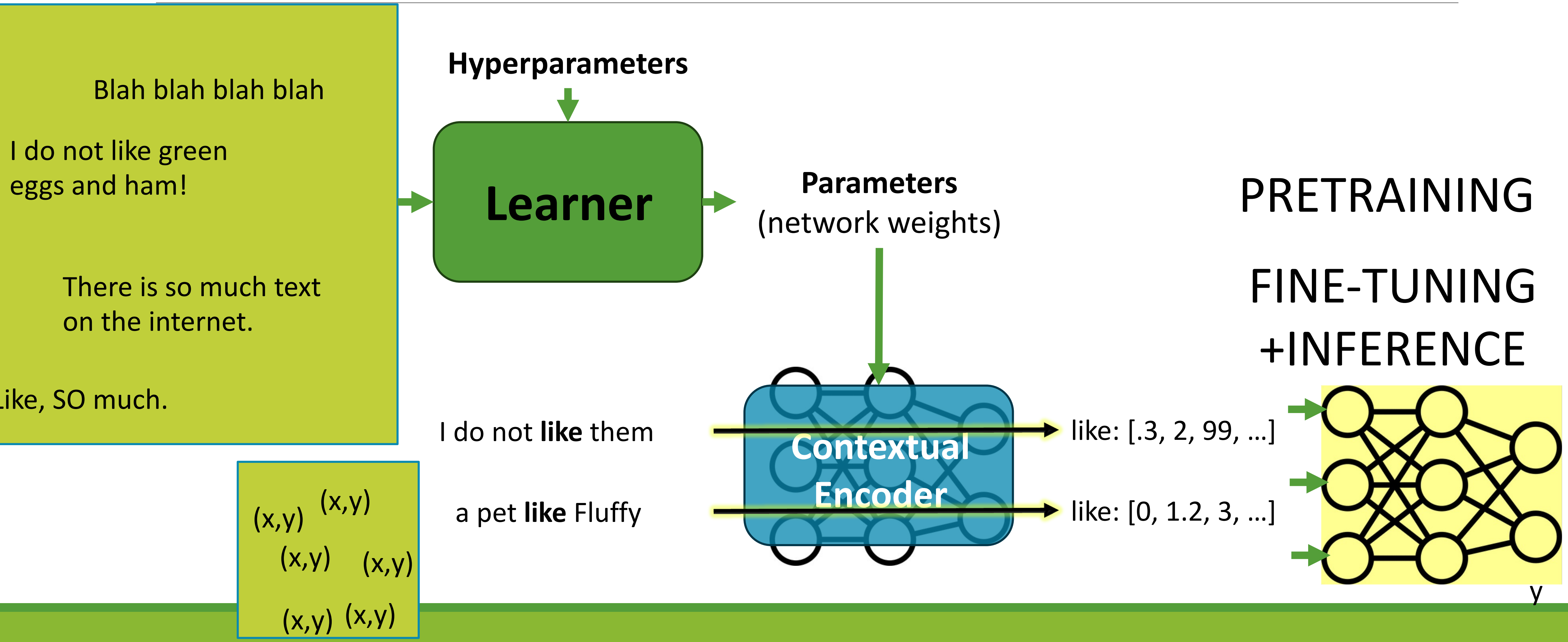
REPRESENTATION

TRAINING+
INFERENCE

## Why Transfer Learning

- Labeled data is often sparse or expensive
- Transfer Learning: taking a pre-trained model and applying it to a new dataset and/or task
- General-purpose **contextualized embeddings** turn out to work well for lots of tasks!
  - Closer to genuine linguistic representations?

# Transfer Learning with LLM



PRETRAINING

FINE-TUNING +INFERENCE

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

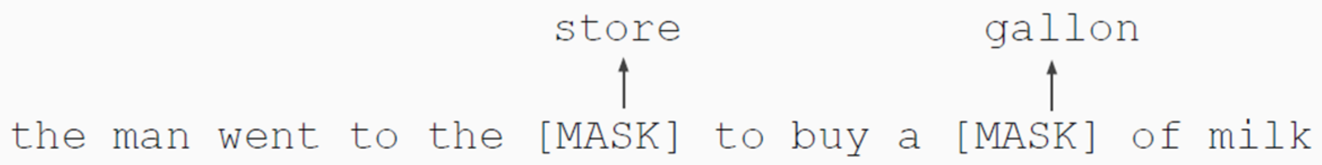
**B**idirectional **E**ncoder **R**epresentations from **T**ransformers

- Idea: if we train a model using a very general task on a **massive** corpus, could we get representations out of that model that are useful for any task?
- Introduced new tasks (masked language modeling, next sentence prediction)

## Masked LM

slides from Devlin et al. (2018)

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



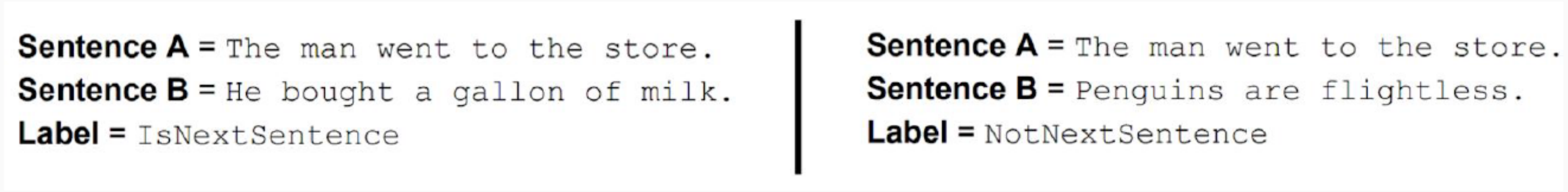
```
the man went to the [MASK] to buy a [MASK] of milk
```

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

## Next Sentence Prediction

slides from Devlin et al. (2018)

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



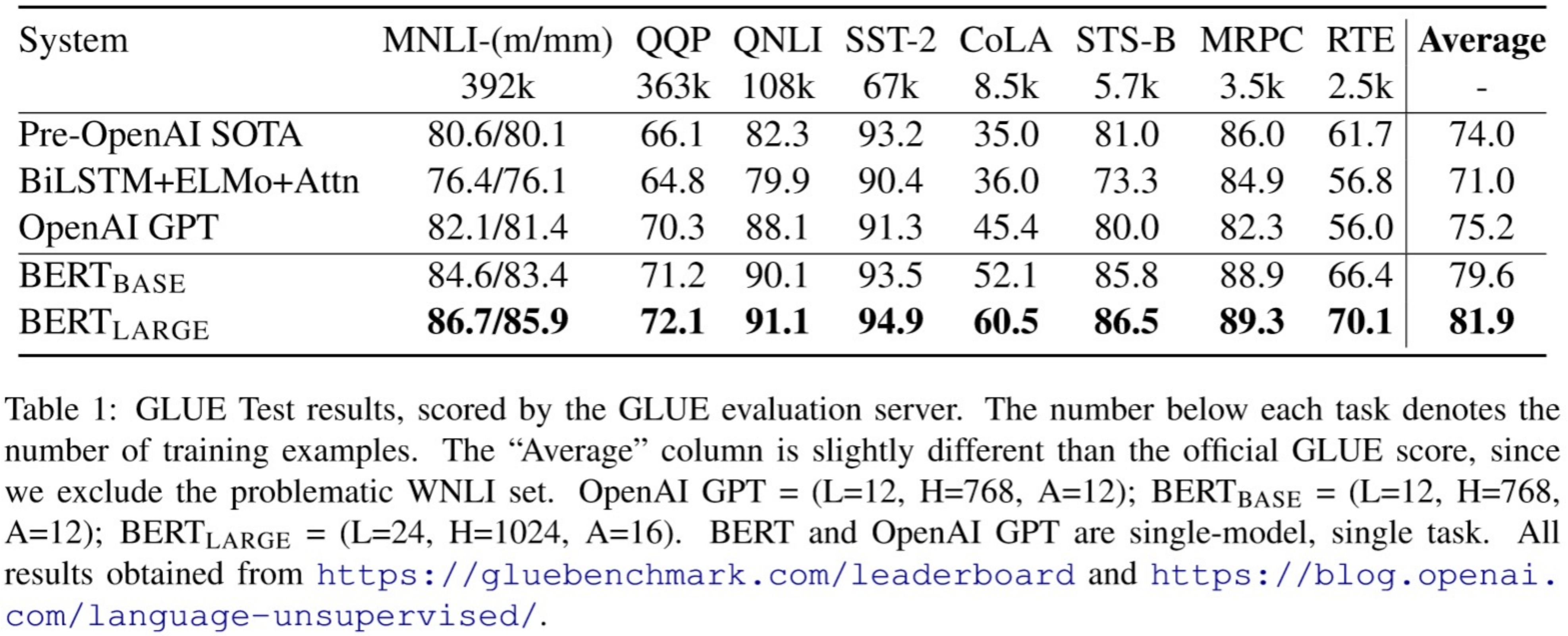
**Sentence A** = `The man went to the store.`
**Sentence B** = `He bought a gallon of milk.`
**Label** = `IsNextSentence`

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

## Experiments

- GLUE: **Textual Inference** (MNLI, RTE, WNLI), **Question Similarity** (QQP), **Question Answering** (QNLI), **Sentiment Analysis** (SST-2), **Grammaticality** (CoLa), **Semantic Similarity** (STS-B, MRPC)
- SQuAD (Question Answering)
- Named Entity Recognition
- SWAG (Adverserial Sentence Prediction)

# Experiments: 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 | 79.9 | 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 |
| BERT$_{\text{BASE}}$ | 84.6/83.4 | 71.2 | 90.1 | 93.5 | 52.1 | 85.8 | 88.9 | 66.4 | 79.6 |
| BERT$_{\text{LARGE}}$ | **86.7/85.9** | **72.1** | **91.1** | **94.9** | **60.5** | **86.5** | **89.3** | **70.1** | **81.9** |

Table 1: GLUE Test results, scored by the GLUE evaluation server. 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. OpenAI GPT = (L=12, H=768, A=12); BERT$_{\text{BASE}}$ = (L=12, H=768, A=12); BERT$_{\text{LARGE}}$ = (L=24, H=1024, A=16). BERT and OpenAI GPT are single-model, single task. All results obtained from https://gluebenchmark.com/leaderboard and https://blog.openai.com/language-unsupervised/.

## Impact

- Most successful early **contextualized word embedding** model
  - "contextualized": a word's representation depends on its context, differs from use to use
- Provided massive performance gains for most tasks in English and other high-resource languages

# Using a contextualized embedding model

- More **pretraining**: you can download a pretrained BERT model and perform additional pretraining on **unlabeled data** in a genre of interest
  - Don’t Stop Pretraining (Gururangan et al. 2020): a good domain adaptation technique
- **Fine-tuning**: You can expose the model to some **labeled data** for your target task to further train the embeddings to suit the task
- **Prediction**: Finally, you can train a classifier on top of BERT, e.g. a linear layer that will use the contextualized embeddings to make a prediction
- **BERTology** is the study of how BERT(-like) models store information about language in various layers, e.g. by controlled comparisons or by using its embeddings for linguistic tasks

## Using a generative LLM

- State-of-the-art **large language models** (LLMs) can now be trained with so much data (+ human feedback) that they can engage in conversations in fluent English.
  - GPT-3.5/ChatGPT, GPT-4, Bard, …
- OpenAI doesn’t release its most advanced GPT models, but you can pay for API access
- Alternatives to fine-tuning for a particular task:
  - **Prompting**: Give an English instruction about want the model to predict (may have to experiment with different ways of phrasing the instruction)
  - **Few-shot learning**: Prompt with a few examples of input/output pairs, then provide test inputs

## Demo

AllenNLP demo of BERT

https://demo.allennlp.org/masked-lm

## Replicate, Extend, etc.

Includes 104 languages

Tensorflow

https://github.com/google-research/bert

Pytorch

https://github.com/huggingface/pytorch-pretrained-BERT

## Recent Transfer Learning Projects

RoBERTa (Liu et al., 2019)

https://github.com/pytorch/fairseq

XLNet (Yang et al., 2019)

https://github.com/zihangdai/xlnet