Deep Learning for NLP

- Oliver Guhr
  I833 SS2019

How is the pace of the lectures so far?

too slow / too fast / just right

a brief recap of the last lecture

Word Representations

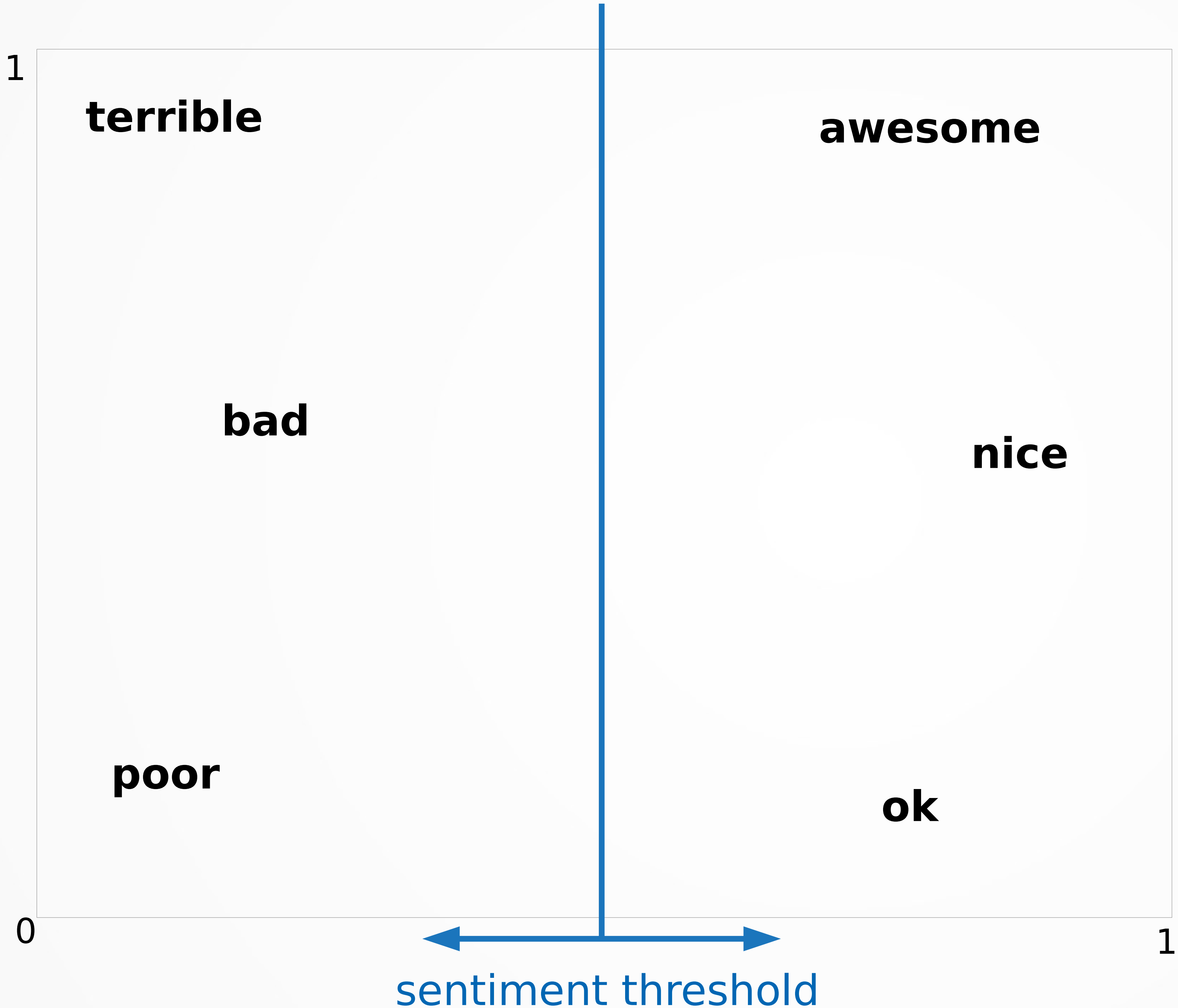
Encoding

Instead of encoding single characters

h	0	0	0	1
e	0	0	1	0
l	0	1	0	0
o	1	0	0	0

You can also encode words, this is also called „Bag-Of-Words (BOW)“

hello	0	0	0	1
my	0	0	1	0
name	0	1	0	0
is	1	0	0	0



Now a network can distinguish between positive and negative words by learning a threshold.

$$v_{ok} = [0.75, 0.15]$$

$$v_{nice} = [0.85, 0.50]$$

$$v_{poor} = [0.15, 0.18]$$

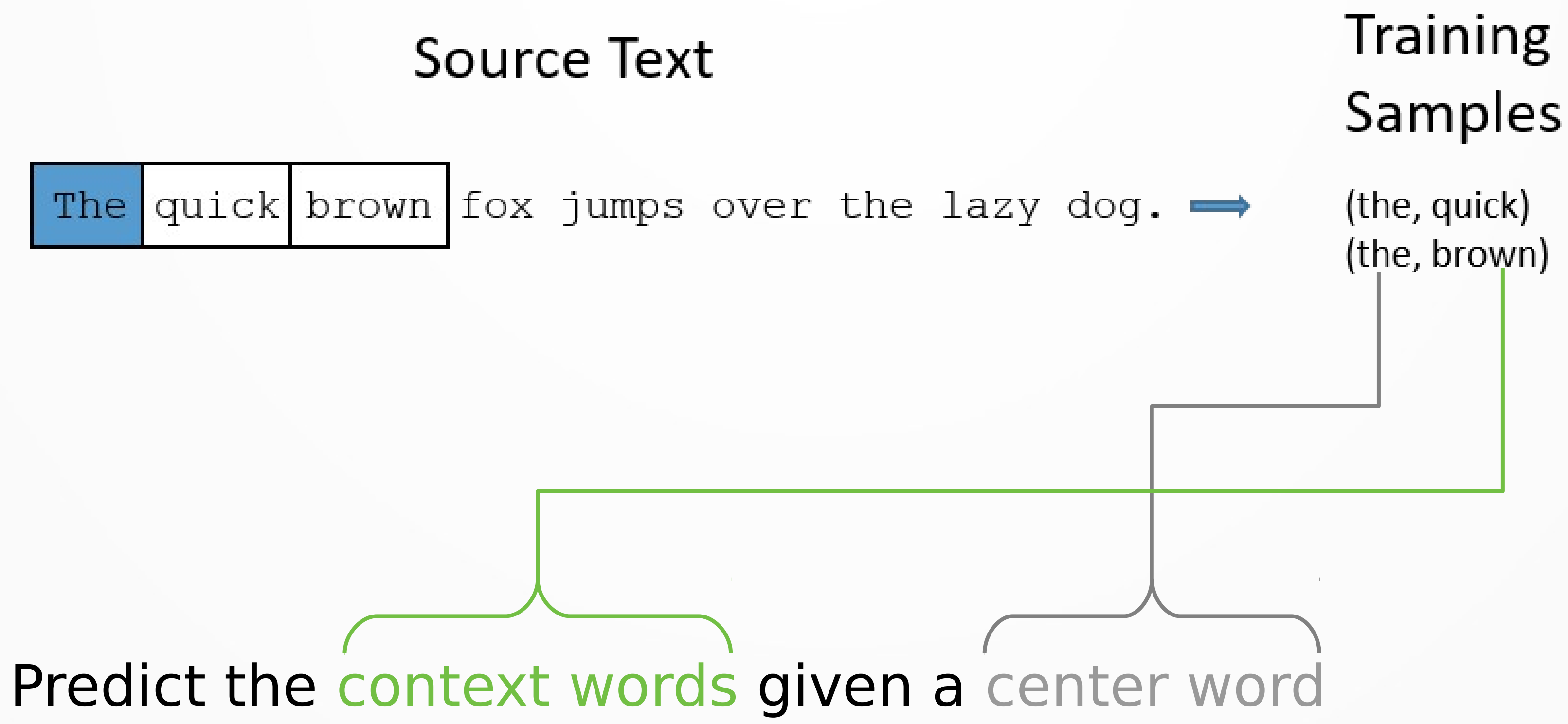
$$v_{terrible} = [0.10, 0.91]$$

Distributional Hypothesis

Words that occur in the same contexts tend to have similar meanings.
Harris (1954)

A word is characterized by the company it keeps.
Firth (1957)

Skip-Gram



McCormick, Word2Vec Tutorial, http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/

Supervised and unsupervised learning

- **Supervised Learning** uses a set of labeled training examples.
  - List of e-mails that are labeled as span / not spam
- **Unsupervised Learning** generates training examples from a plain –unlabeled- text corpus, so the training becomes a supervised problem.
  - Predict context words (Skip Gram)
  - Predict center words (Cbow)
  - Predict masked words (Bert)

Preprocessing

- Format you text to be predictable and analyzable
- It often has a significant impact on the performance
- Depending on the domain and your model different steps may be required
- For example:
  - Cleaning not useful characters and word
  - Transform words into a standardized form
  - Clipping your data to equal length

Scores: Accuracy

- $$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn}$$

- $$\text{Accuracy} = \frac{\text{number correctly predicted samples}}{\text{total number of samples}}$$

Scores: F1

$$precision = \frac{true\ \ positives}{true\ \ positives + false\ \ positives}$$

$$recall = \frac{true\ \ positives}{true\ \ positives + false\ \ negatives}$$

$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$

How to optimize:

1) create a train/test split
2) Train your model (start with a simple model!)
3) measure its performance
4) optimize your model
5) Go to 2 :)

Identify offensive language

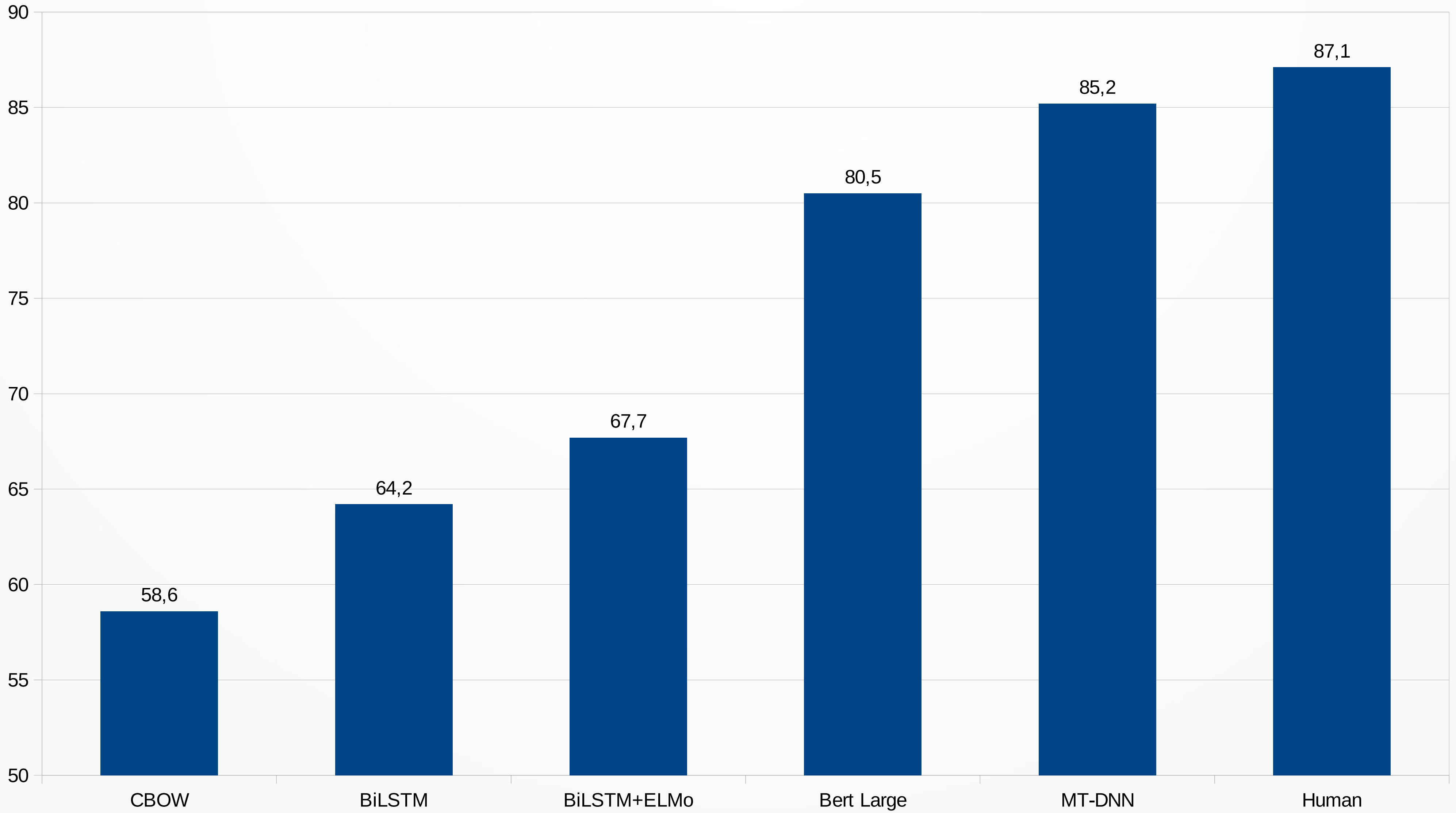
using word vectors and FastText

Goal for Today

- Look we look at deep language models
  - How do they perform?
  - How do they work?
  - Some of the basic ideas behind those models.
  - How to use these models.
  - Applications :)

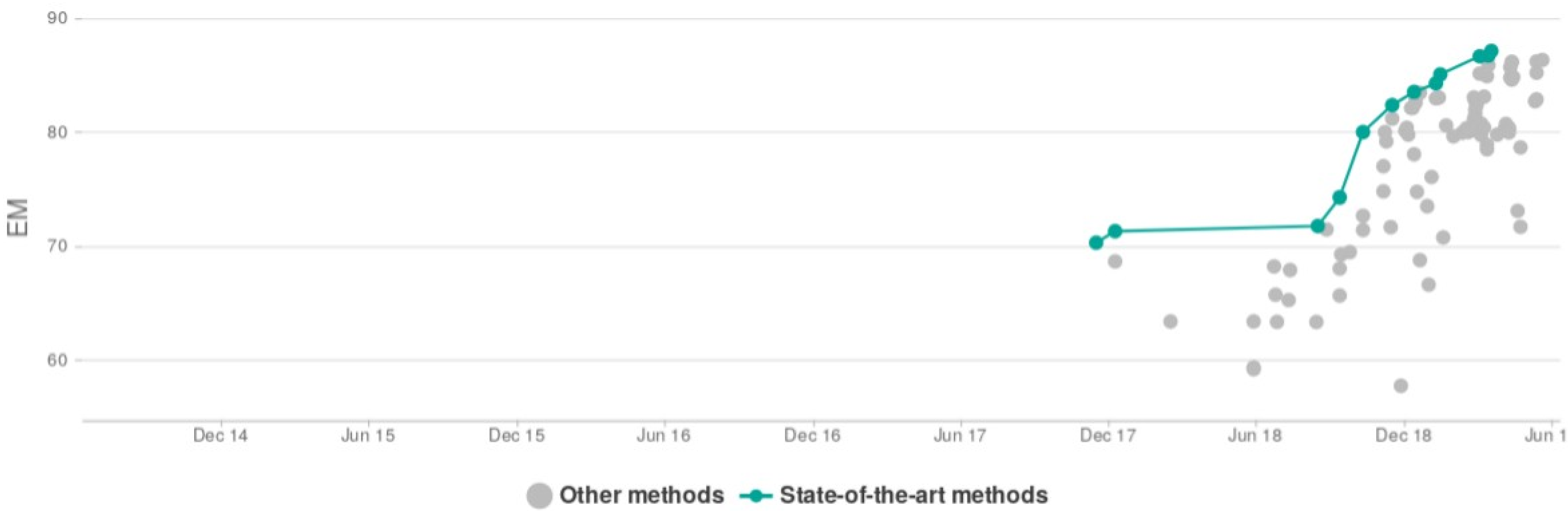
Deep Language Models

GLUE Benchmark Results



GLUE Leaderboard: https://gluebenchmark.com/leaderboard

SQUAD 2.0



Source: https://paperswithcode.com/sota/question-answering-on-squad20

Deep Language Models

- In 2018 several Ideas led to new models
  - Semi-supervised Sequence Learning Andrew Dai, Quoc Le
  - ELMo Peters et al.
  - ULMFiT Howard, Ruder
  - OpenAI Transformer Radford, Narasimhan, Salimans, Sutskever
  - Transformer Vaswani et al.

Deep Language Models

- Google's BERT (October 2018)
  - BERT: Pre-training of Deep Bidirectional Transformers for Language Understa
nding
- Google/CMU's Transformer-XL (January 2019)
  - Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context
- OpenAI's GPT (June, 2018)
  - Improving Language Understanding by Generative Pre-Training
- OpenAI's GPT-2 (February, 2019)
  - Language Models are Unsupervised Multitask Learners

Model Parameters

Model	Parameters
Medium LSTM	10 Million
ELMo	90 Million
GTP	110 Million
Bert Base	110 Million
Bert Large	340 Million
GTP-2	1500 Million

BERT

Bidirectional

Encoder

Representations from

Transformers

Bert

- What can you do with Bert?
- Some applications:
  - Named Entity Recognition
  - Text Classification
  - Fact Checking
  - Text Summarization
  - Text Generation
  - Question Answering (Full Text and Multiple Choice)
  - Translation

Bert

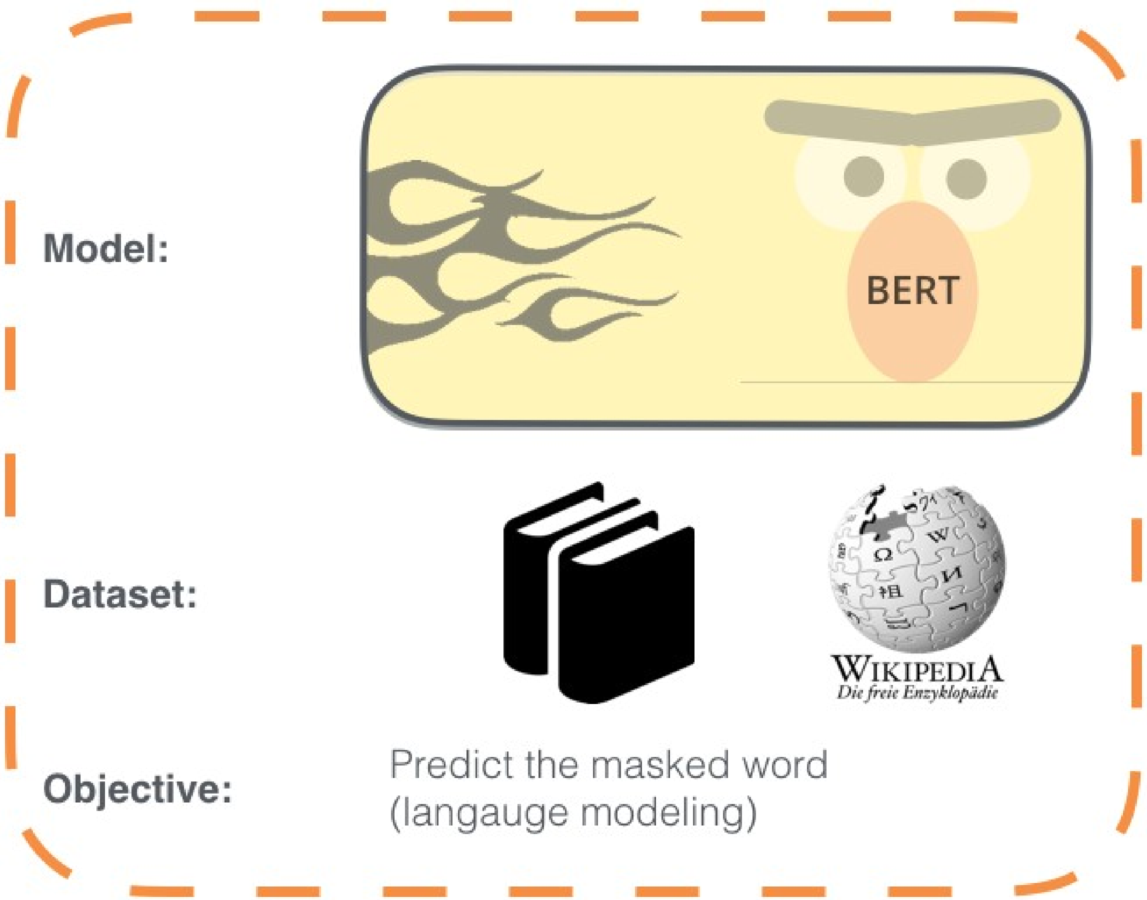
- Training Process
  - Pre-train a model on plain text
  - Choose a task specific labeled data set
  - Retrain the model with this data set

- Use the same pre trained model for all tasks
    - Classification
    - Named Entity Recognition
    - Question Answering etc.

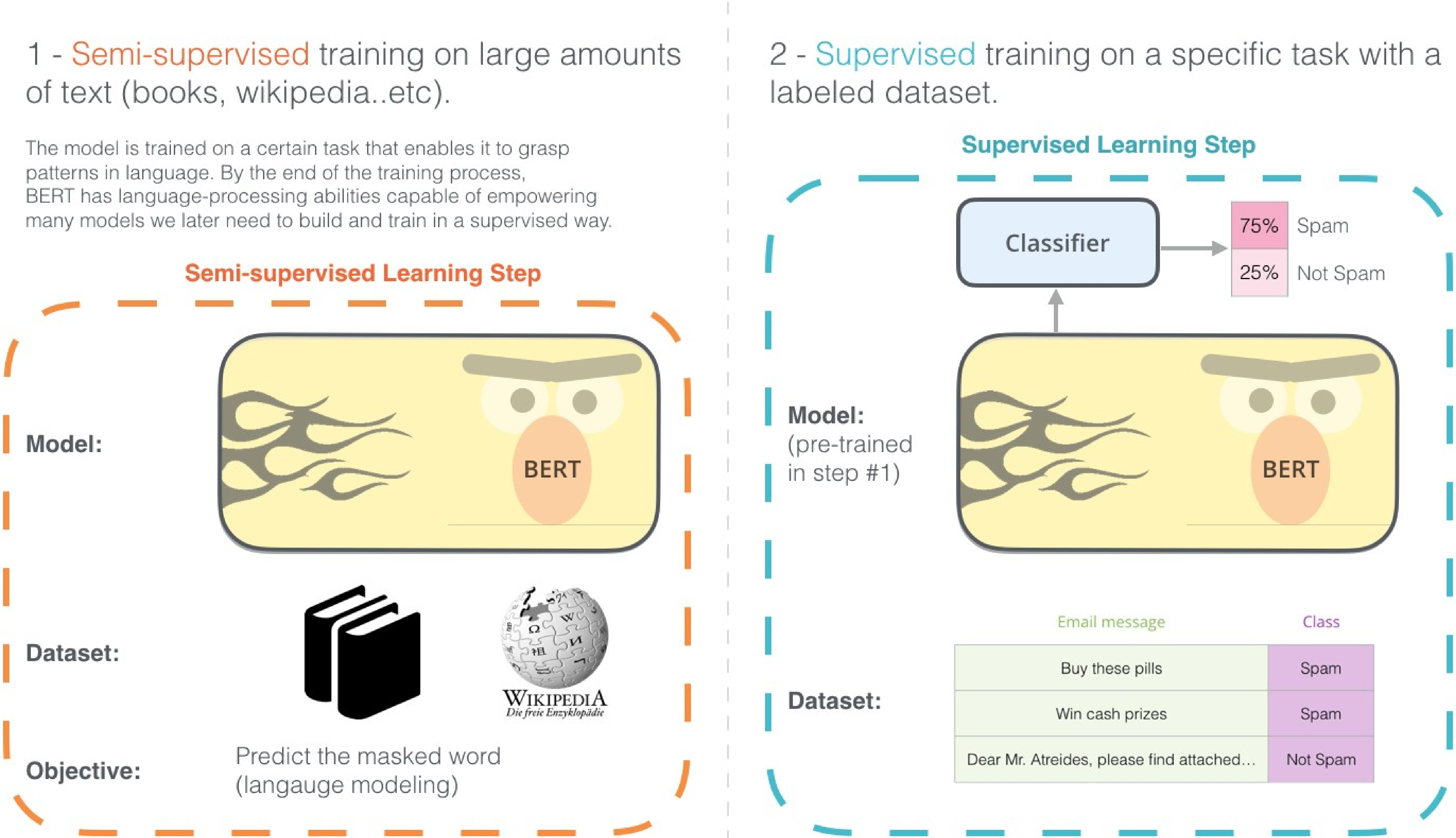
1 - Semi-supervised training on large amounts of text (books, wikipedia..etc).

The model is trained on a certain task that enables it to grasp patterns in language. By the end of the training process, BERT has language-processing abilities capable of empowering many models we later need to build and train in a supervised way.

Semi-supervised Learning Step

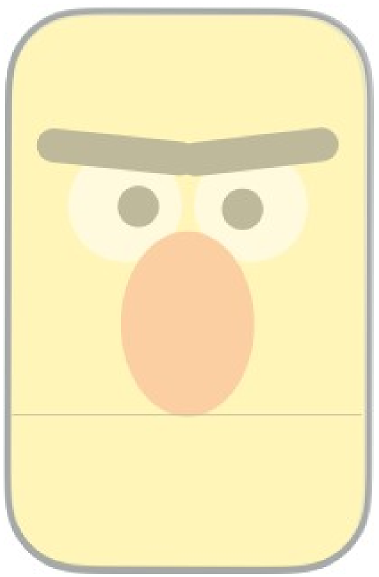


The Illustrated BERT, Jay Alammar: http://jalammar.github.io/illustrated-bert/

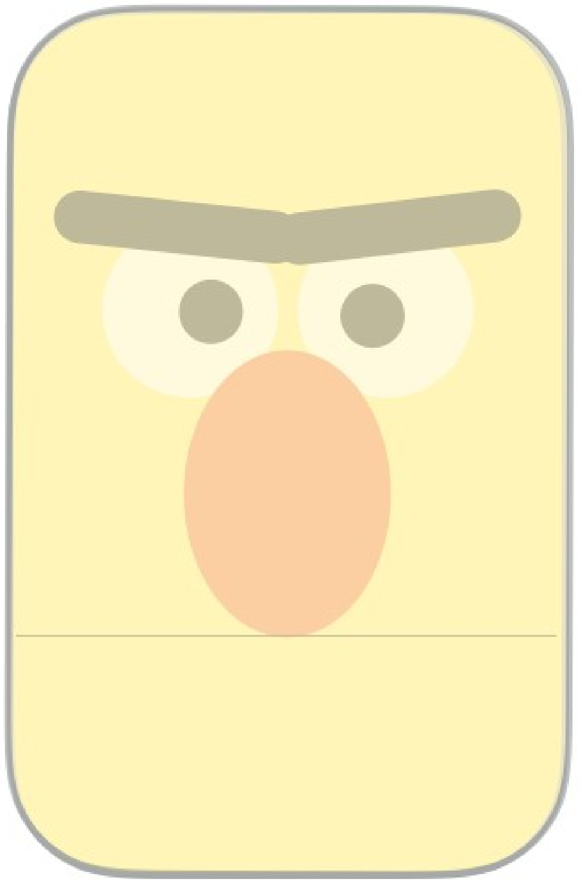


The Illustrated BERT, Jay Alammar: http://jalammar.github.io/illustrated-bert/

Two pre-trained sizes



$\text{BERT}_{\text{BASE}}$



$\text{BERT}_{\text{LARGE}}$

The Illustrated BERT, Jay Alammar: http://jalammar.github.io/illustrated-bert/

Pre Trained Bert

- English
  - BERT-Large, Uncased (Whole Word Masking): 24-layer, 1024-hidden, 16-heads, 340M parameters
  - BERT-Large, Cased (Whole Word Masking): 24-layer, 1024-hidden, 16-heads, 340M parameters
  - BERT-Base, Uncased: 12-layer, 768-hidden, 12-heads, 110M parameters
  - BERT-Large, Uncased: 24-layer, 1024-hidden, 16-heads, 340M parameters
  - BERT-Base, Cased: 12-layer, 768-hidden, 12-heads , 110M parameters
  - BERT-Large, Cased: 24-layer, 1024-hidden, 16-heads, 340M parameters
- Multi Language
  - BERT-Base, Multilingual Cased (New, recommended): 104 languages, 12-layer, 768-hidden, 12-heads, 110M parameters
  - BERT-Base, Multilingual Uncased (Orig, not recommended) (Not recommended, use Multilingual Cased instead): 102 languages, 12-layer, 768-hidden, 12-heads, 110M parameters
- Chinese
  - BERT-Base, Chinese: Chinese Simplified and Traditional, 12-layer, 768-hidden, 12-heads, 110M parameters

Pre Trained Bert

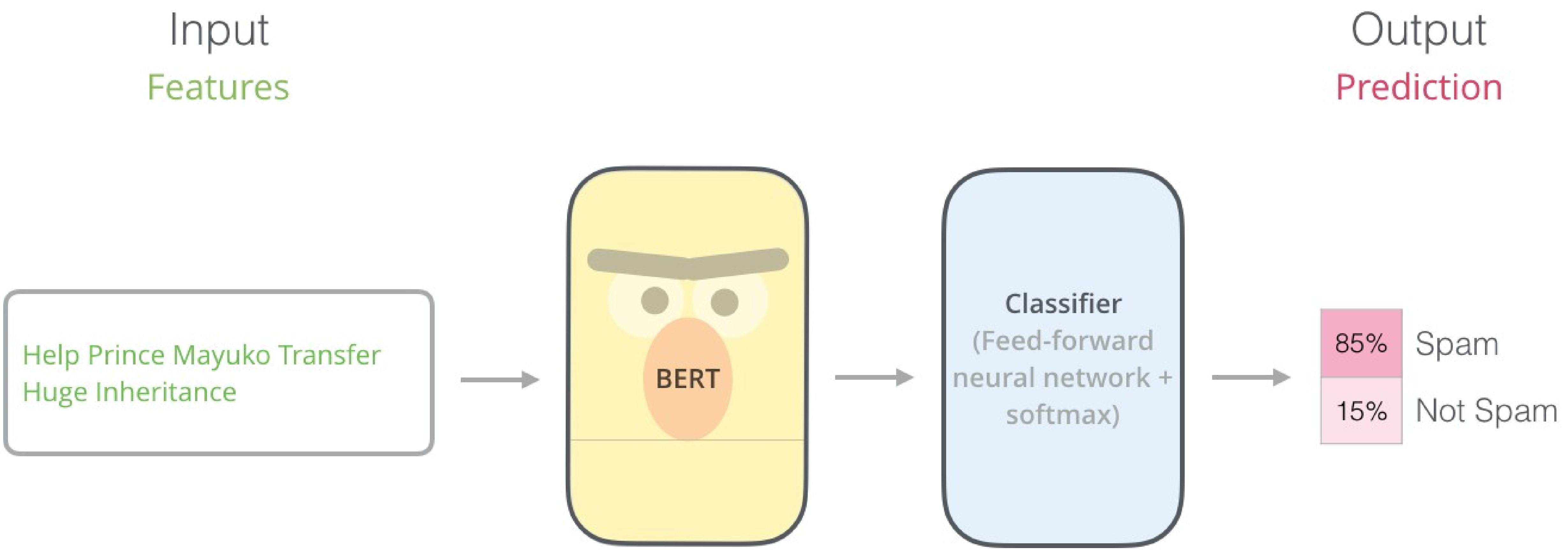
Explained on the next slides

- English
  - BERT-Large, Uncased (Whole Word Masking): $\boxed{\text{24-layer, 1024-hidden, 16-heads,}}$ 340M parameters
  - BERT-Large, Cased (Whole Word Masking): 24-layer, 1024-hidden, 16-heads, 340M parameters
  - BERT-Base, Uncased: 12-layer, 768-hidden, 12-heads, 110M parameters
  - BERT-Large, Uncased: 24-layer, 1024-hidden, 16-heads, 340M parameters
  - BERT-Base, Cased: 12-layer, 768-hidden, 12-heads , 110M parameters
  - BERT-Large, Cased: 24-layer, 1024-hidden, 16-heads, 340M parameters
- Multi Language
  - BERT-Base, Multilingual Cased (New, recommended): 104 languages, 12-layer, 768-hidden, 12-heads, 110M parameters
  - BERT-Base, Multilingual Uncased (Orig, not recommended) (Not recommended, use Multilingual Cased instead): 102 languages, 12-layer, 768-hidden, 12-heads, 110M parameters
- Chinese
  - BERT-Base, Chinese: Chinese Simplified and Traditional, 12-layer, 768-hidden, 12-heads, 110M parameters

Bert

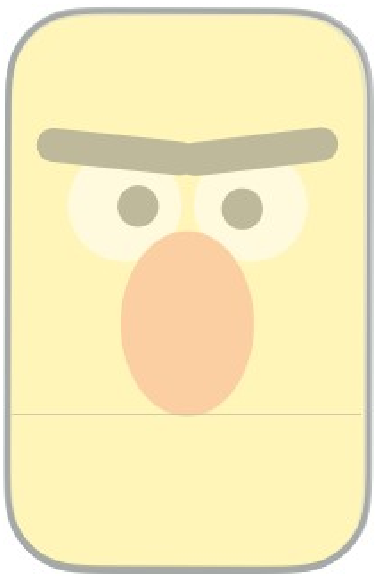
- Issues with non English models
  - One model for 102 languages
  - Pre-trained on Wikipedia content
  - English Models are pre-trained on books and Wikipedia
  - Train you own language model will cost ca. 500$
  - Collecting this data for German language would be a great task fo a research seminar :)

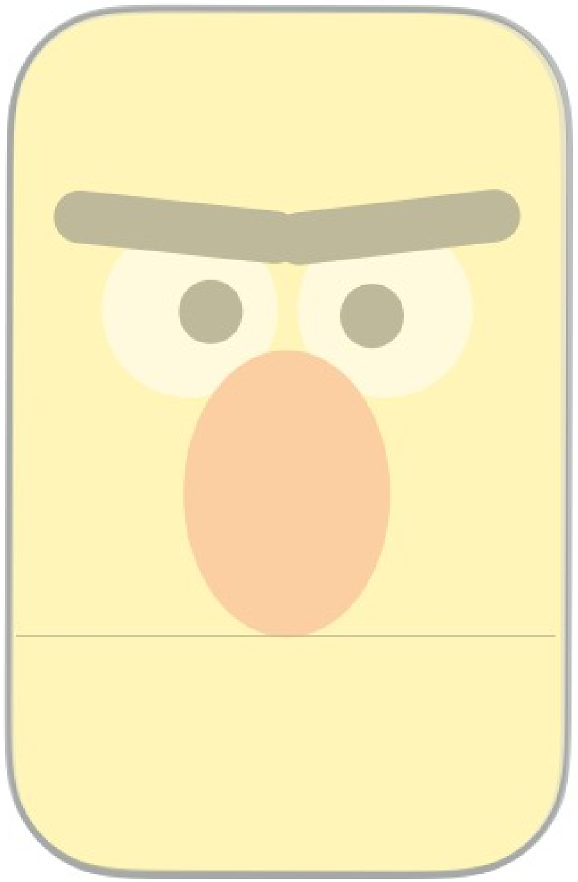
Text Classification



The Illustrated BERT, Jay Alammar: http://jalammar.github.io/illustrated-bert/

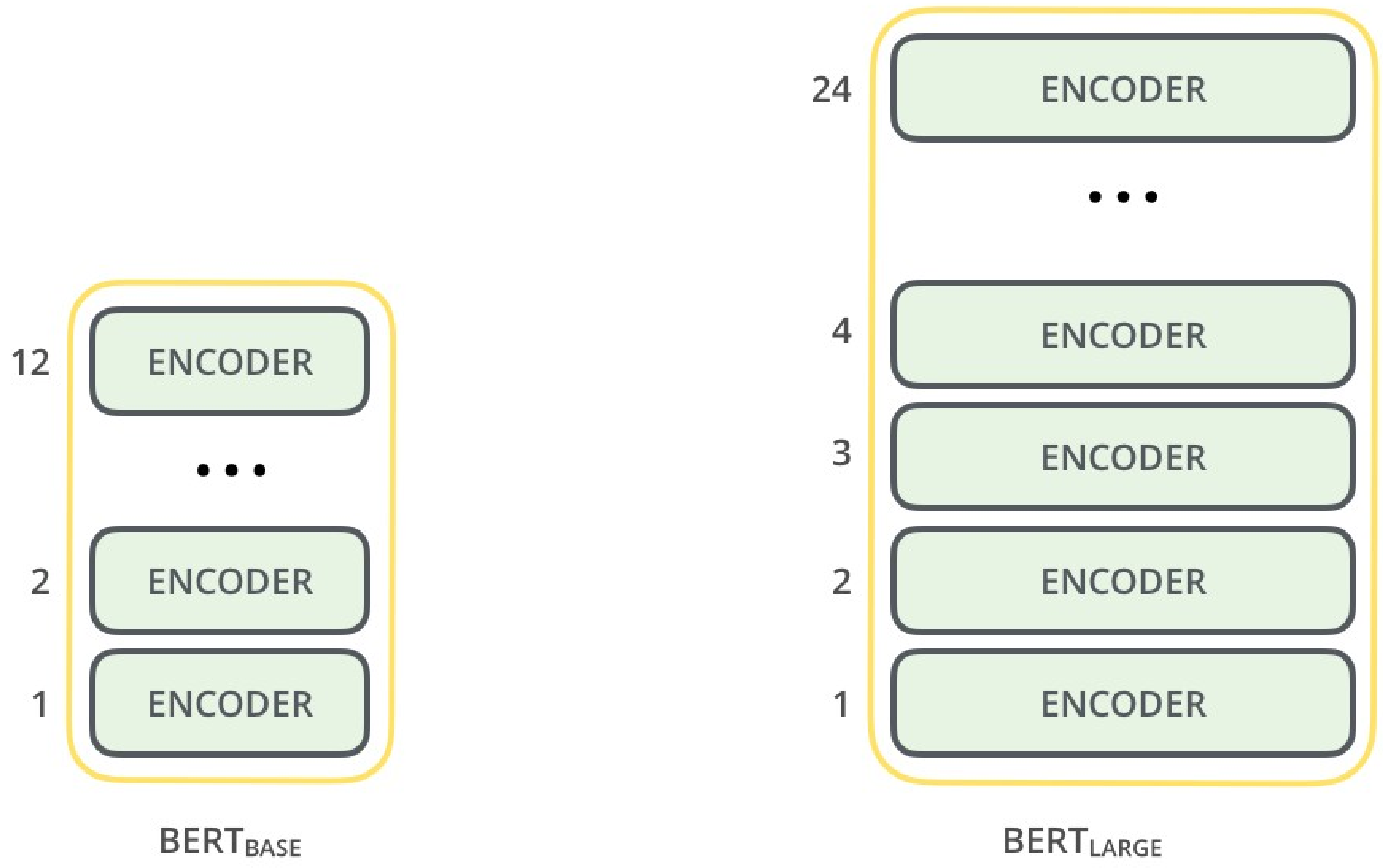
Two sizes





The Illustrated BERT, Jay Alammar: http://jalammar.github.io/illustrated-bert/

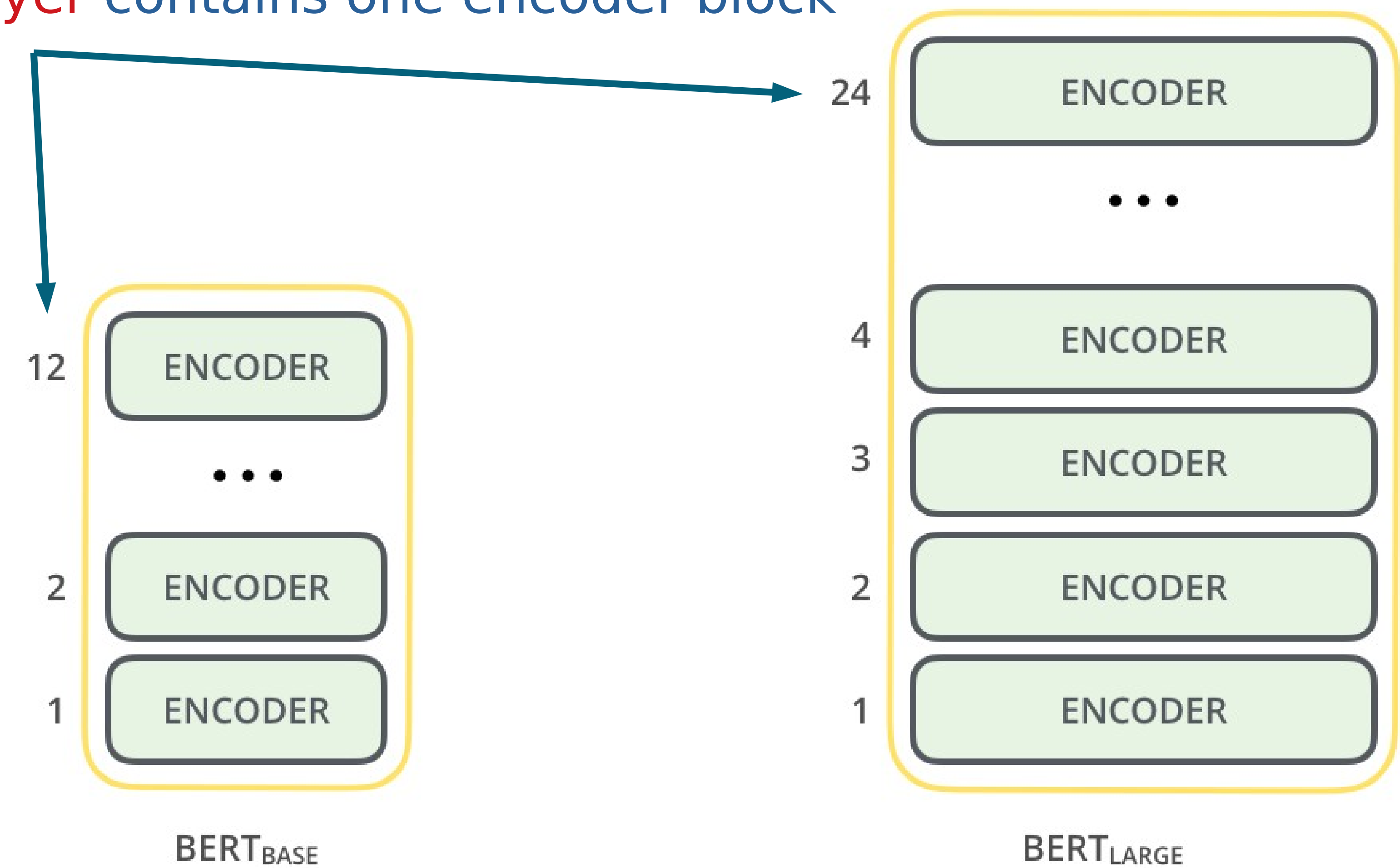
Bert Model



The Illustrated BERT, Jay Alammar: http://jalammar.github.io/illustrated-bert/

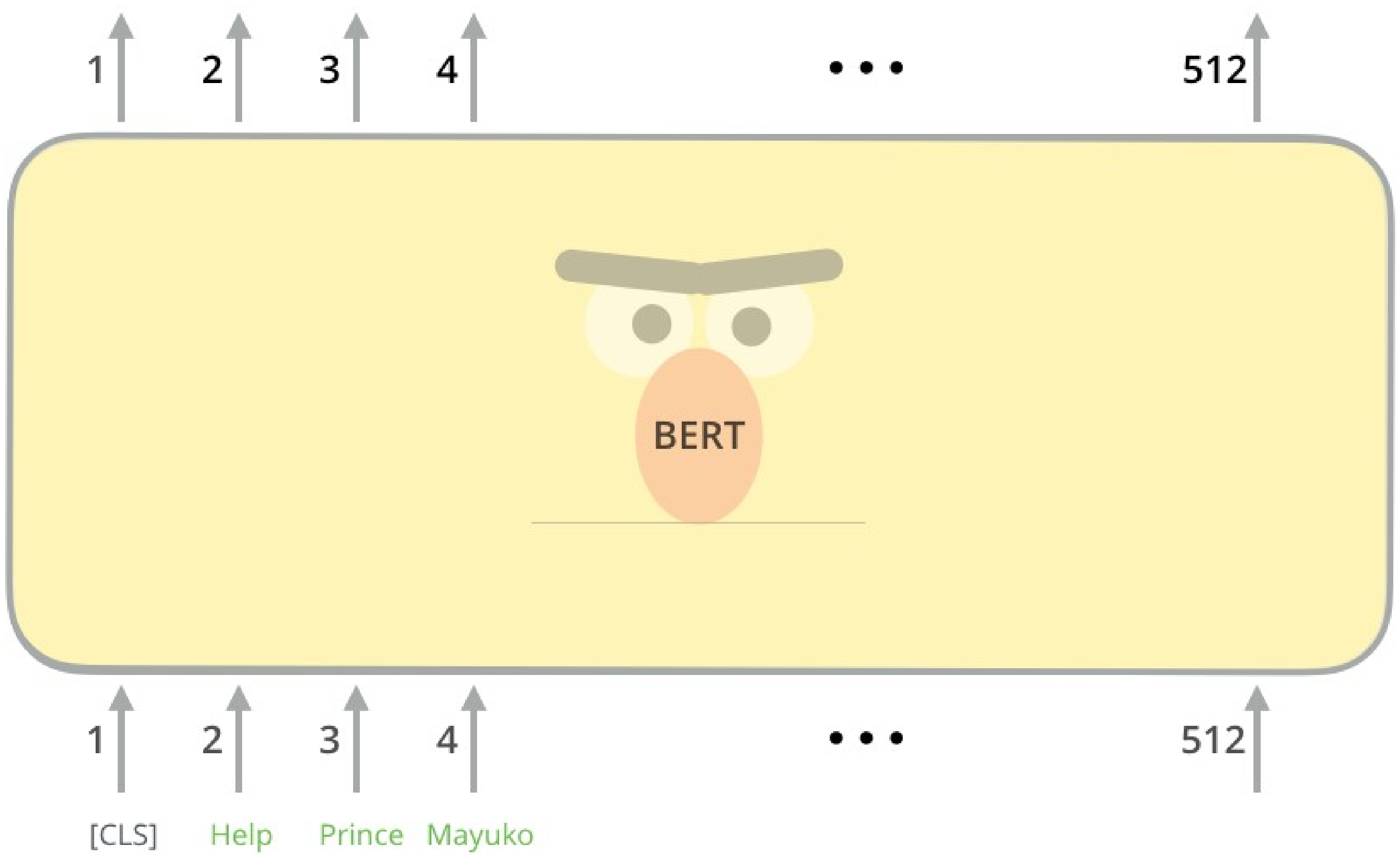
Bert Model

Each layer contains one encoder block



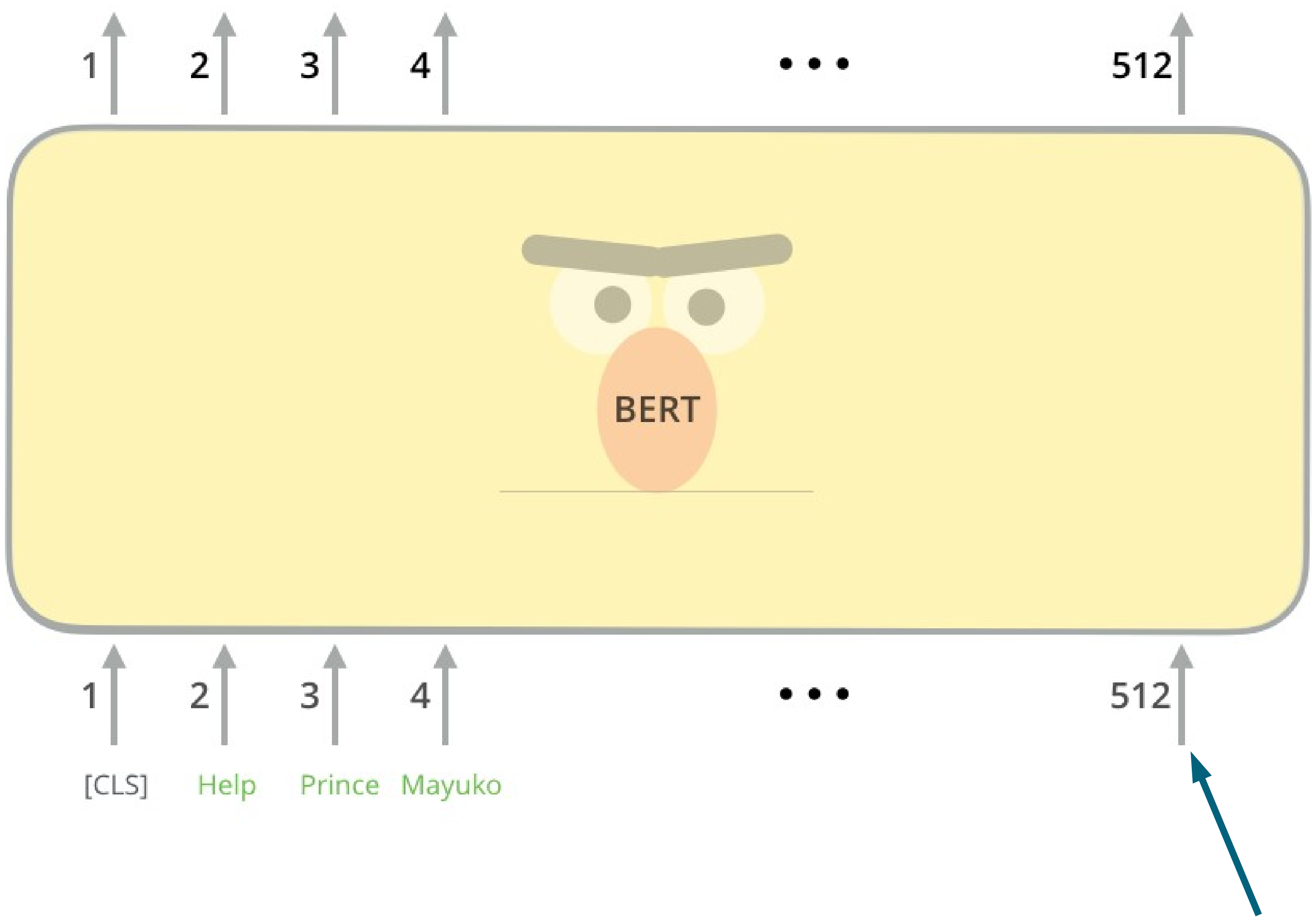
The Illustrated BERT, Jay Alammar: http://jalammar.github.io/illustrated-bert/

How to process sequences?



The Illustrated BERT, Jay Alammar: http://jalammar.github.io/illustrated-bert/

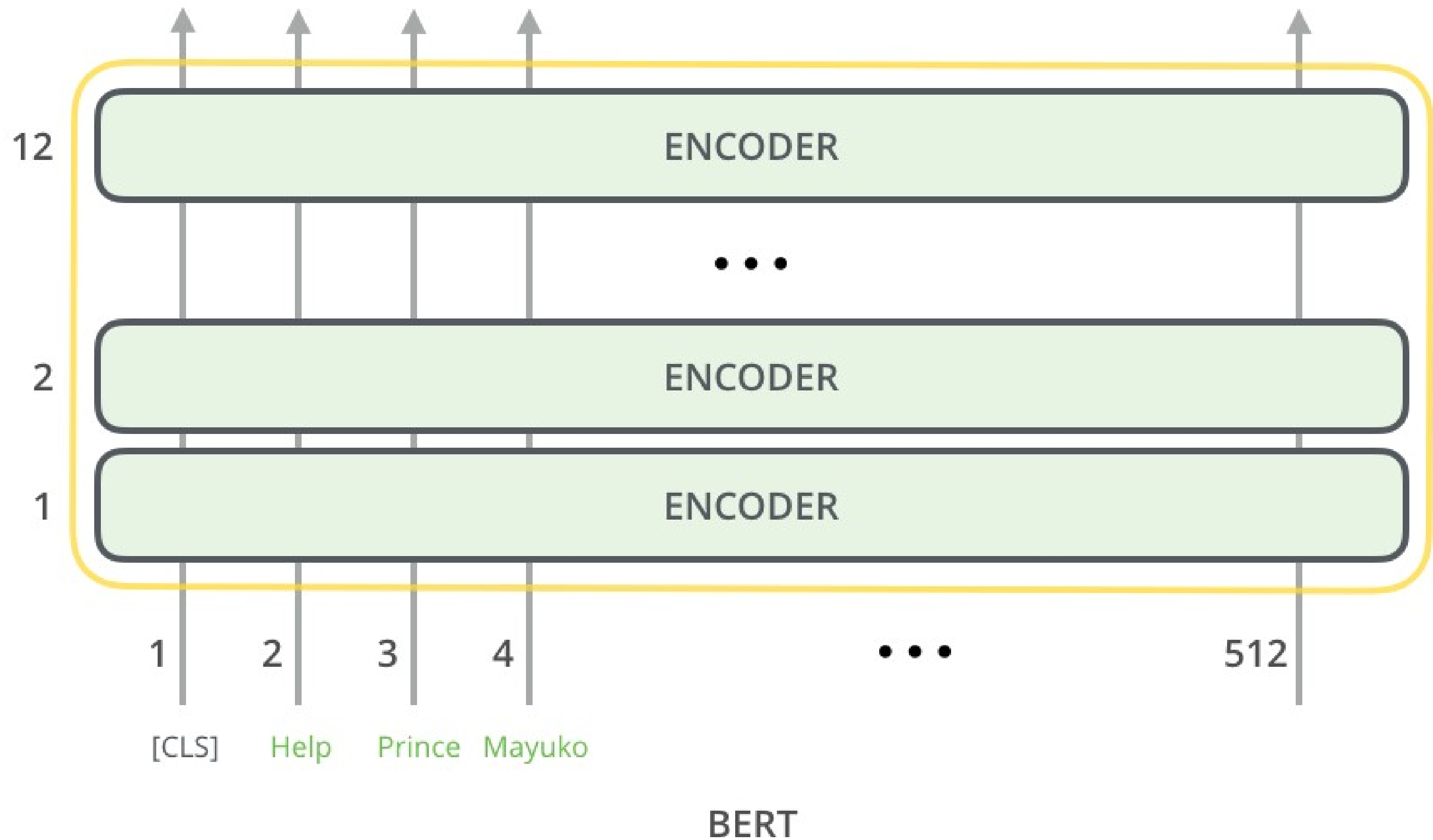
How to process sequences?



This Bert model can process sequences up to 512 tokens.

The Illustrated BERT, Jay Alammar: http://jalammar.github.io/illustrated-bert/

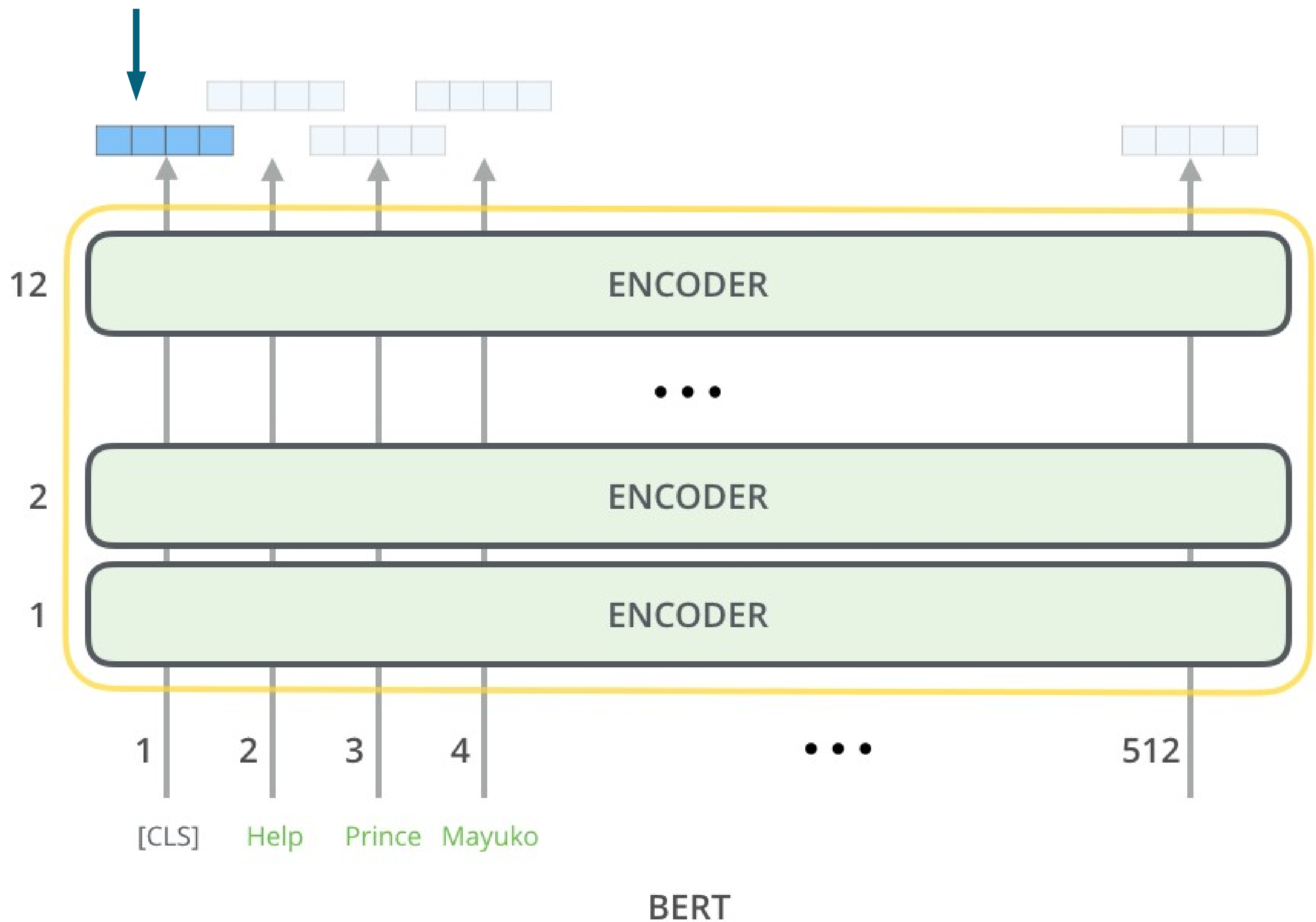
Bert



The Illustrated BERT, Jay Alammar: http://jalammar.github.io/illustrated-bert/

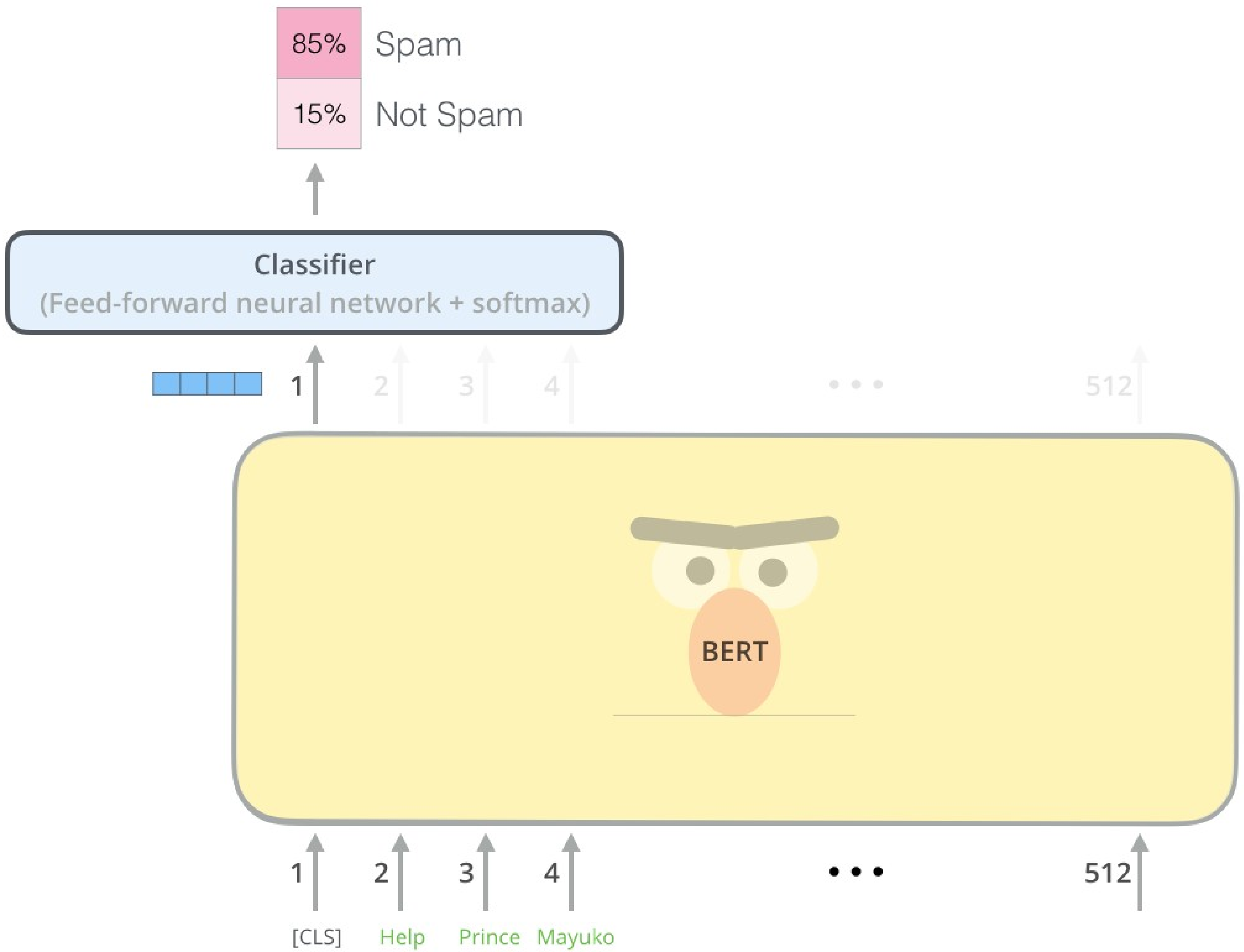
Bert

Each token generates a vector with the length of the hidden size.



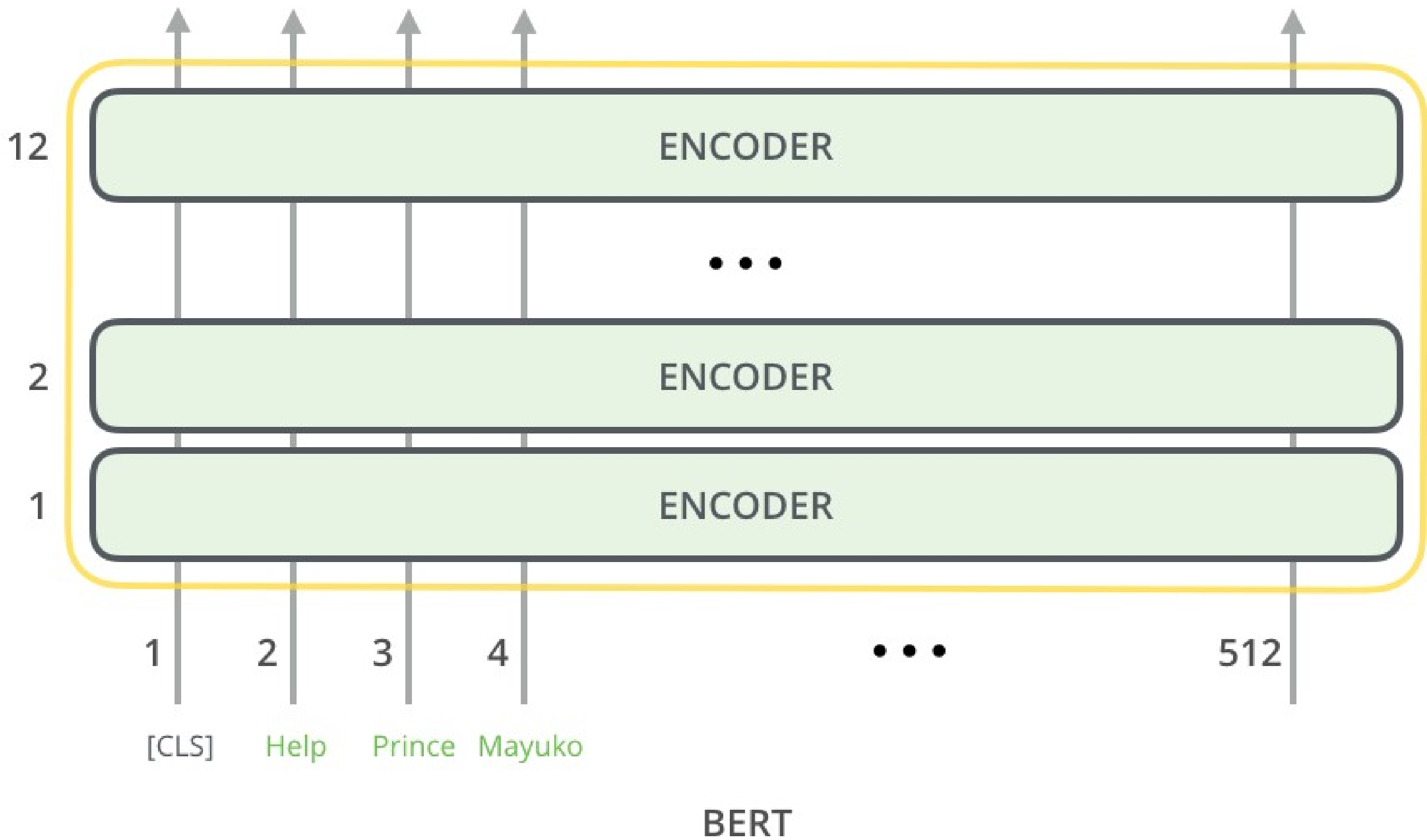
The Illustrated BERT, Jay Alammar: http://jalammar.github.io/illustrated-bert/

Classification with Bert

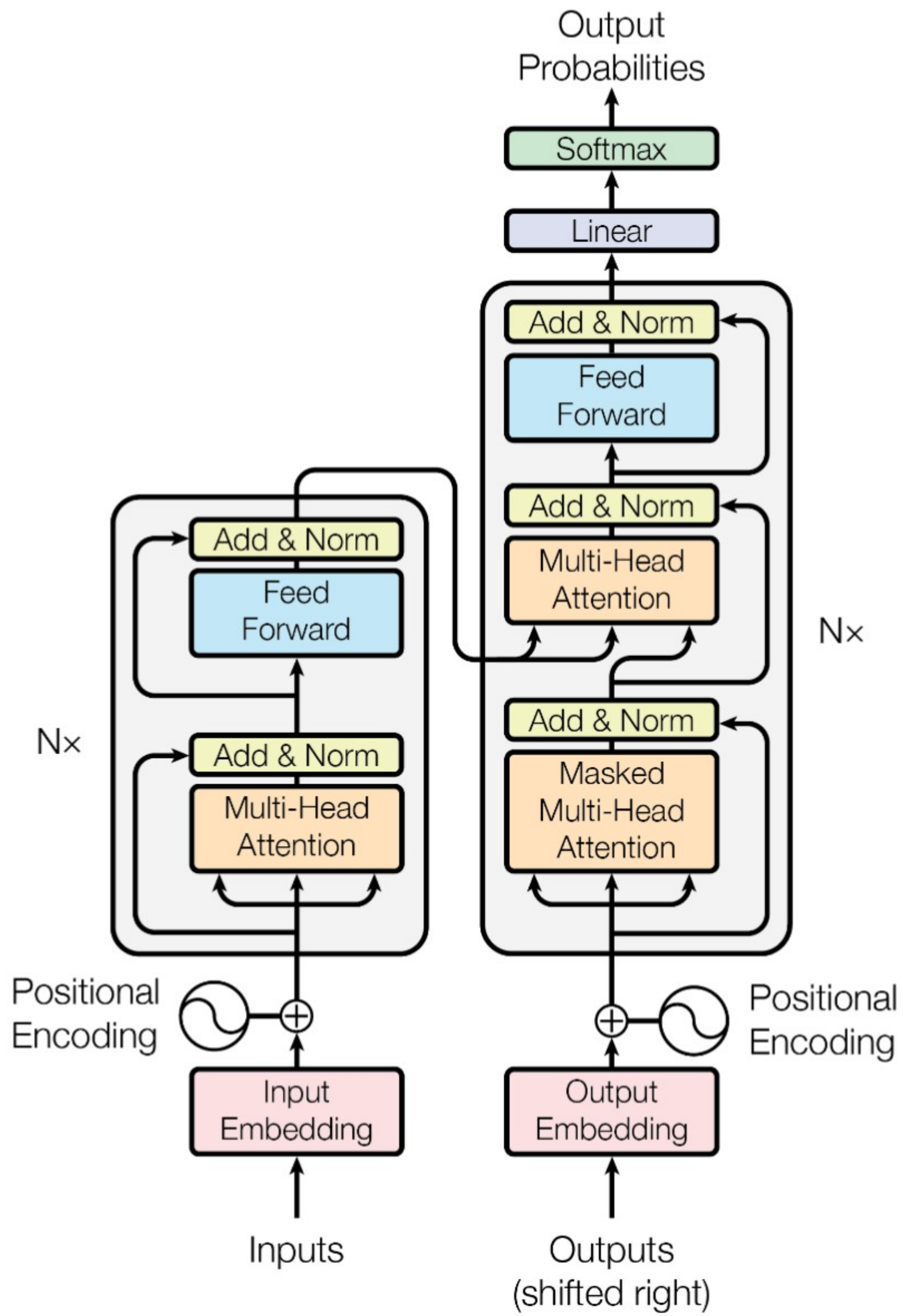


Attention and Transformer

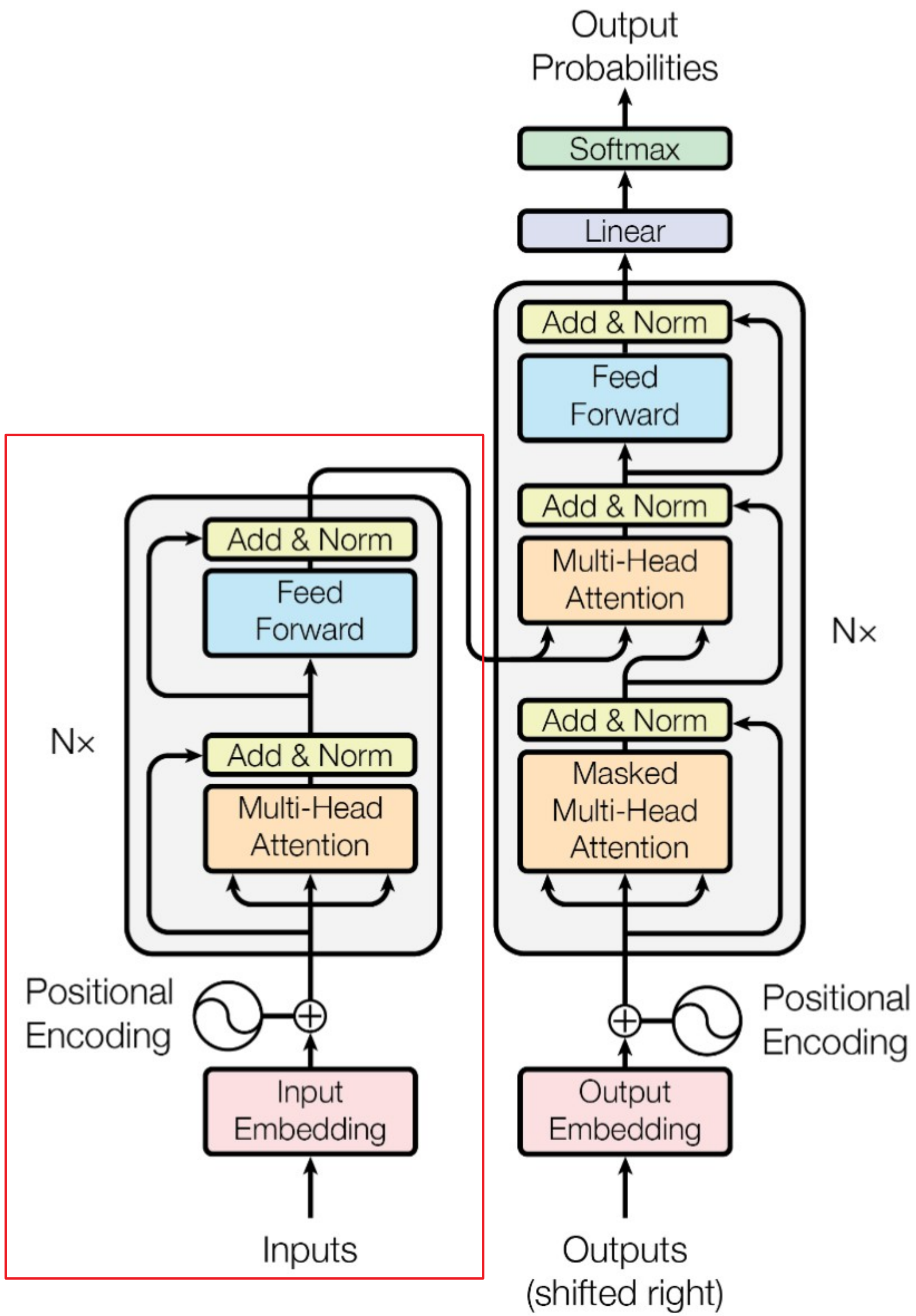
Berts Encoder



The Illustrated BERT, Jay Alammar: http://jalammar.github.io/illustrated-bert/

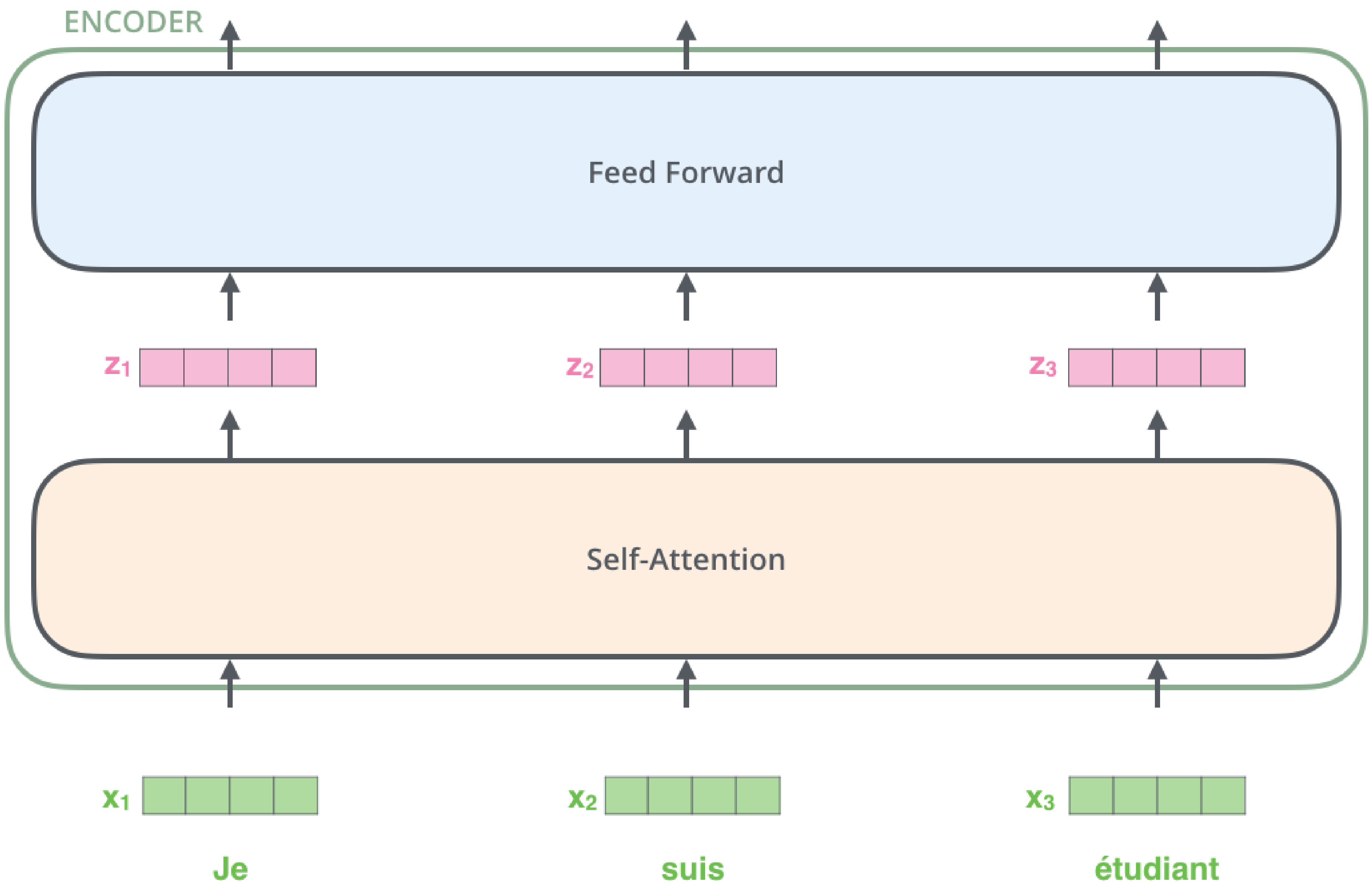


Attention Is All You Need, Vaswani et al. https://arxiv.org/abs/1706.03762



Attention Is All You Need, Vaswani et al. https://arxiv.org/abs/1706.03762

Transfromer



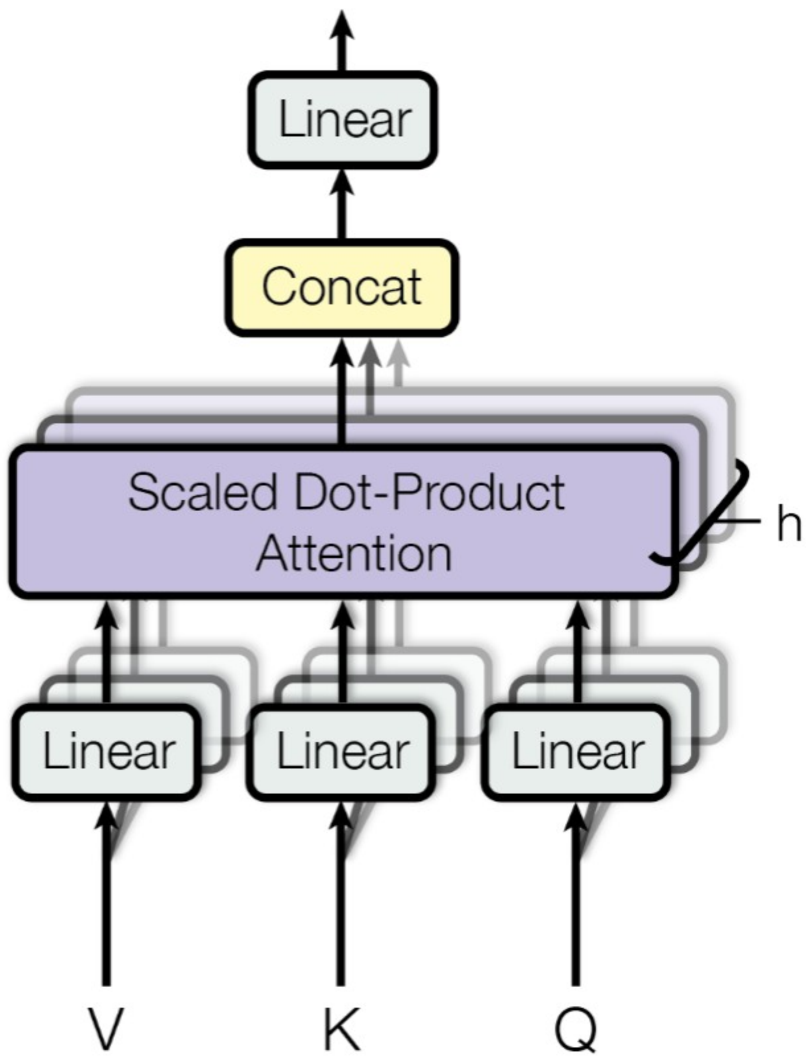
The Illustrated BERT, Jay Alammar: http://jalammar.github.io/illustrated-bert/

Transformer Attention

$$A(\underbrace{Q}_{\text{Query}},\underbrace{K}_{\text{Key}},\underbrace{V}_{\text{Value}})=softmax(QK^{T})V$$

Take the current word or token, find the most similar Key and return the corresponding value.

Transformer: Multi Head Attention

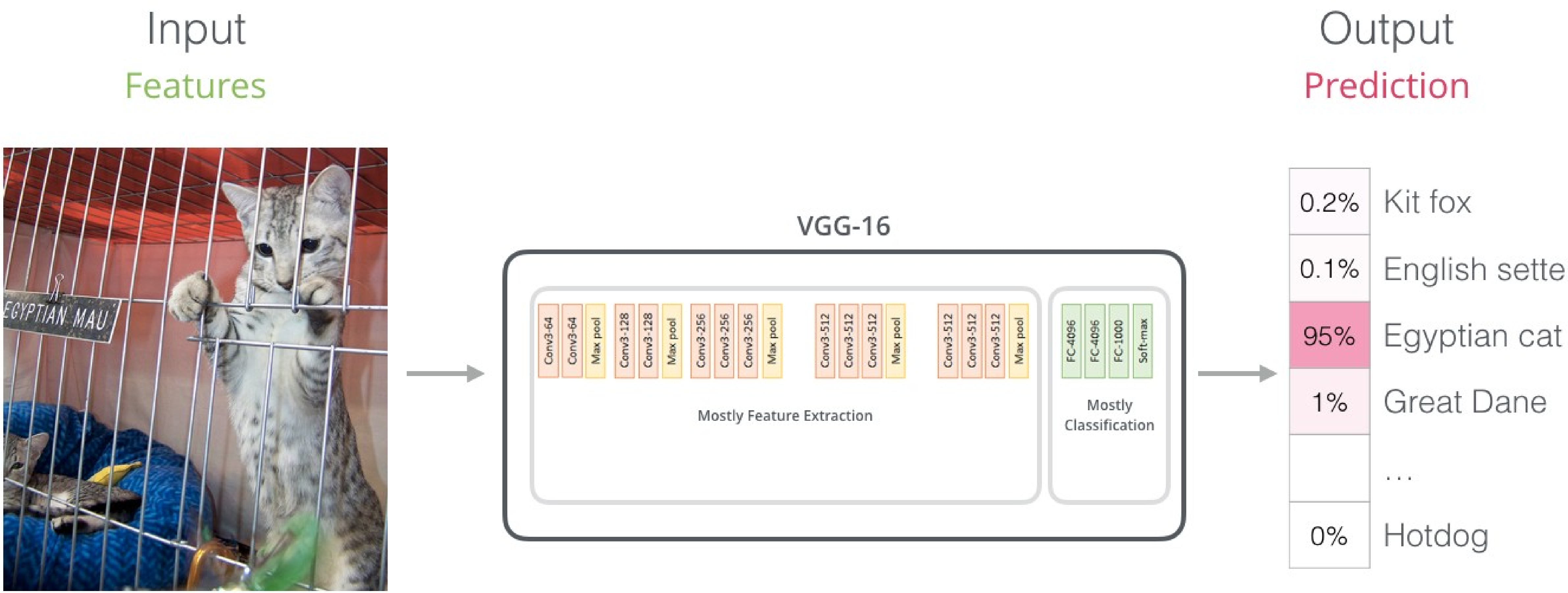


Attention Is All You Need, Vaswani et al. https://arxiv.org/abs/1706.03762

Transformer

- Paper
  - Attention is all you need. Vaswani et al.
- Good Read
  - Jay Alammars The Illustrated Transformer
- Conference Talk:
  - Attention is all you need attentional neural network models by Łukasz Kaiser

Similarity to Conv Nets

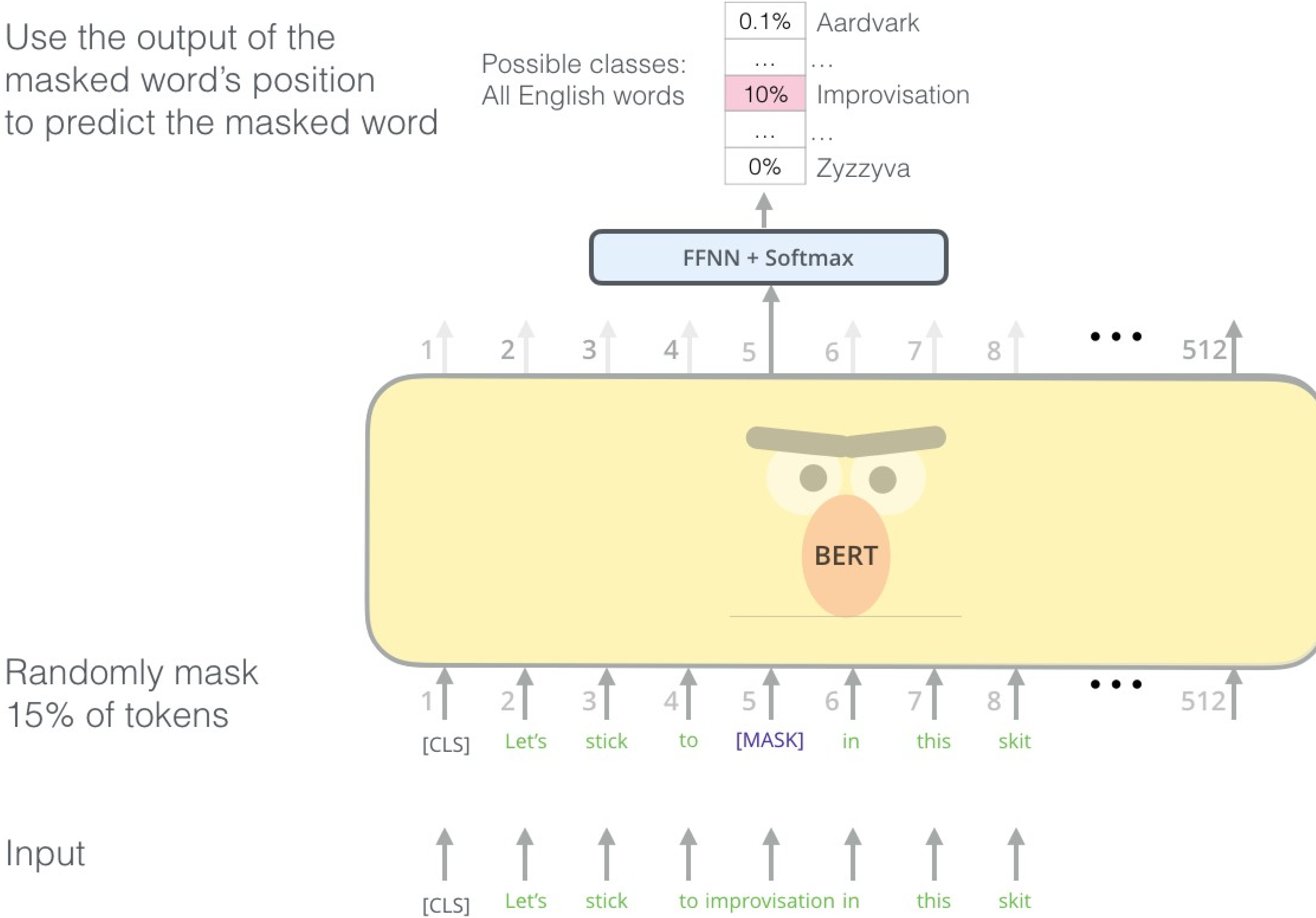


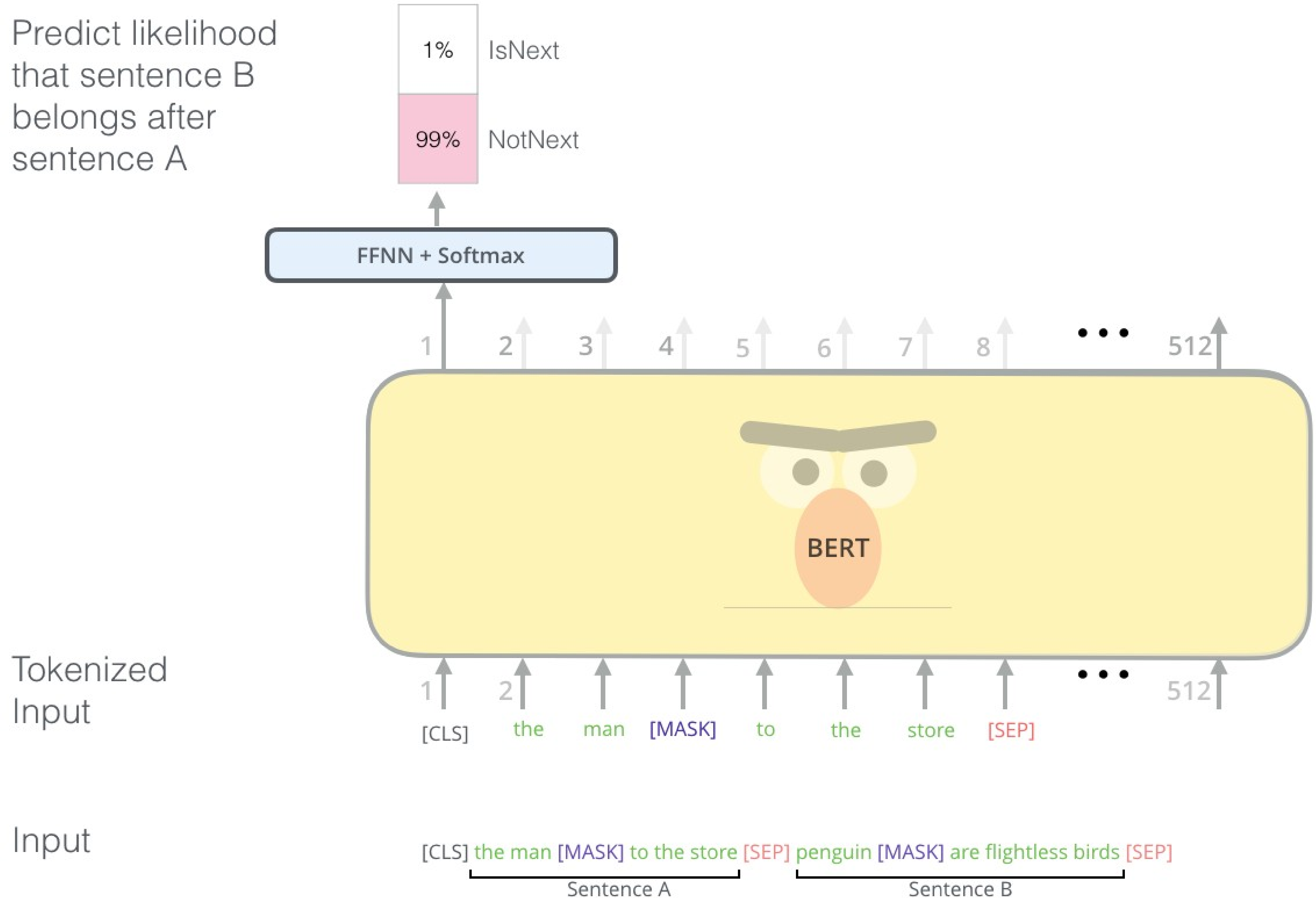
The Illustrated BERT, Jay Alammar: http://jalammar.github.io/illustrated-bert/

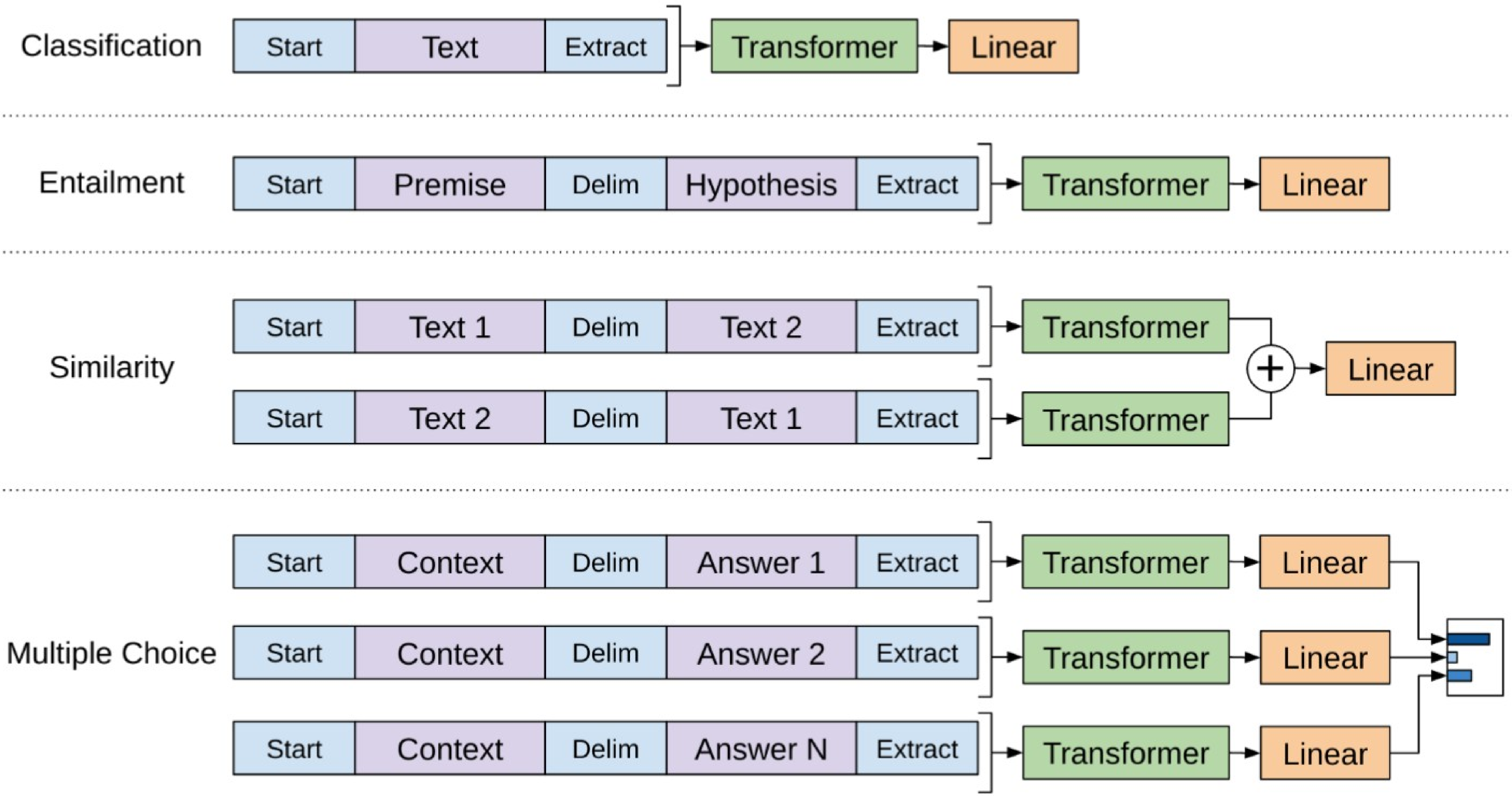
Training Bert

Pre Training Bert

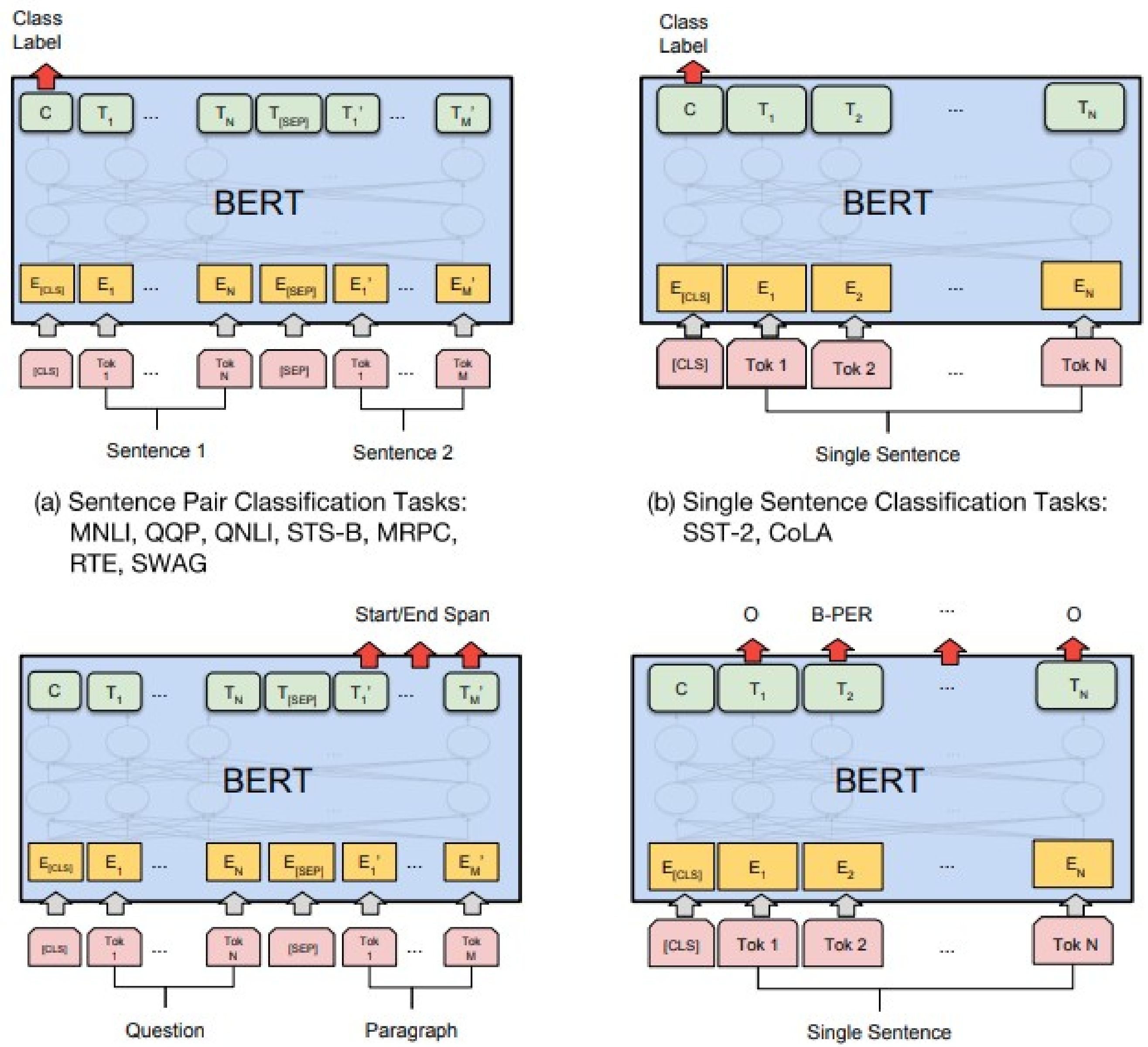
- pre trained models are also called **language models**
- Compared to FastText Berts language model can distinguish between contexts
  - "river bank" vs "financial bank"
- To create them, Bert used two methods:
  - Task One: Mask Words
  - Task Two: Next Sentence Prediction


Use the output of the masked word's position to predict the masked word
Possible classes: All English words
0.1% Aardvark
... ...
10% Improvisation
... ...
0% Zyzzyva
FFNN + Softmax
1 2 3 4 5 6 7 8 ... 512
BERT
Randomly mask 15% of tokens
1 2 3 4 5 6 7 8 ... 512
[CLS] Let's stick to [MASK] in this skit
Input
[CLS] Let's stick to improvisation in this skit


Predict likelihood that sentence B belongs after sentence A
1% IsNext
99% NotNext
FFNN + Softmax
1 2 3 4 5 6 7 8 512
BERT
Tokenized Input
1 2 512
[CLS] the man [MASK] to the store [SEP]
Input
[CLS] the man [MASK] to the store [SEP] penguin [MASK] are flightless birds [SEP]
Sentence A
Sentence B



Improving Language Understandingby Generative Pre-Training, Radford et. al



(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

(b) Single Sentence Classification Tasks: SST-2, CoLA

(c) Question Answering Tasks: SQuAD v1.1

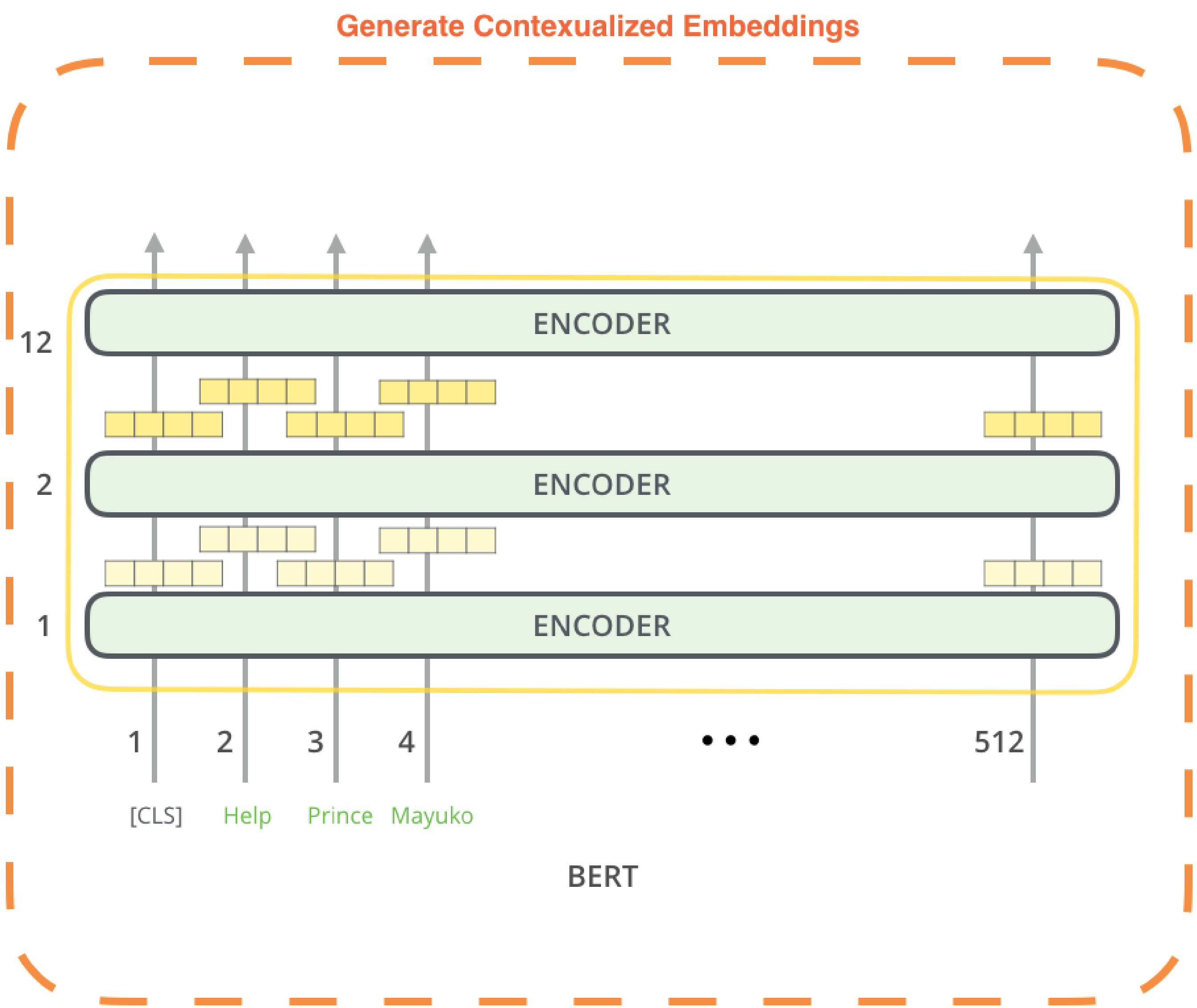
(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

Improving Language Understandingby Generative Pre-Training, Radford et. al

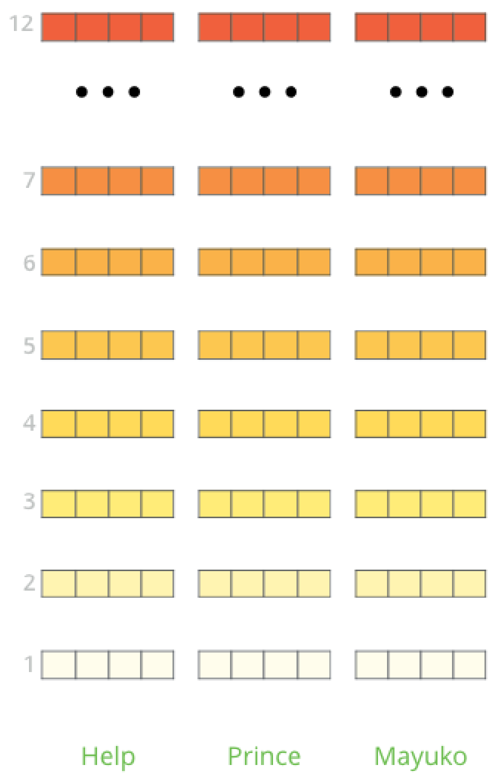
Traing Bert

- Pre Training takes 14 days on a TPUv2 (500$)
- Bert Large Models (24 Layers) can only be trained on TPUs
- Fine-tuning a model with 1GB of text takes serveral hours on a single GPU (1080 / 2080)

Bert as Embedding



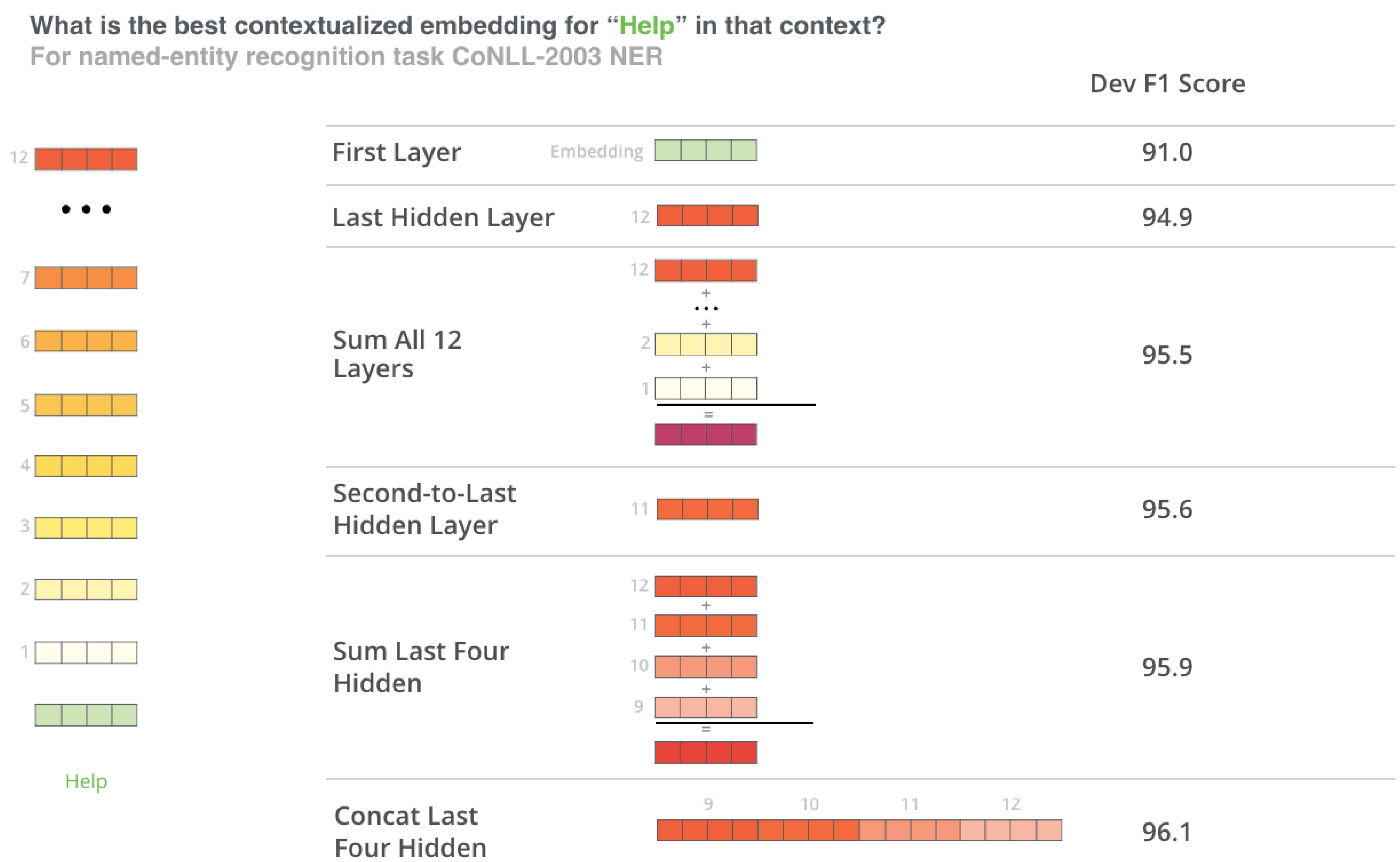
The output of each encoder layer along each token's path can be used as a feature representing that token.



But which one should we use?

The Illustrated BERT, Jay Alammar: http://jalammar.github.io/illustrated-bert/

Bert as Embedding



The Illustrated BERT, Jay Alammar: http://jalammar.github.io/illustrated-bert/

Summary

- Pre-Trained on unlabeled data, fine tune on domain specific data
- Better language understanding
  - Context matters!
- Better results with less data
  - Latest hotness MT-DNN Liu et al.
  - No need for vast amounts of training data

Identify offensive language

using word vectors and FastText

Identify offensive language

~~using word vectors and FastText~~

using Bert

WHK Jobs

- Do cutting edge deep learning:
  - Dialog Systems aka Chat Bots
  - Speech Recognition
  - Speech Synthesis
  - Text Classification / Generation
  - Build Alexa and Mycroft skills