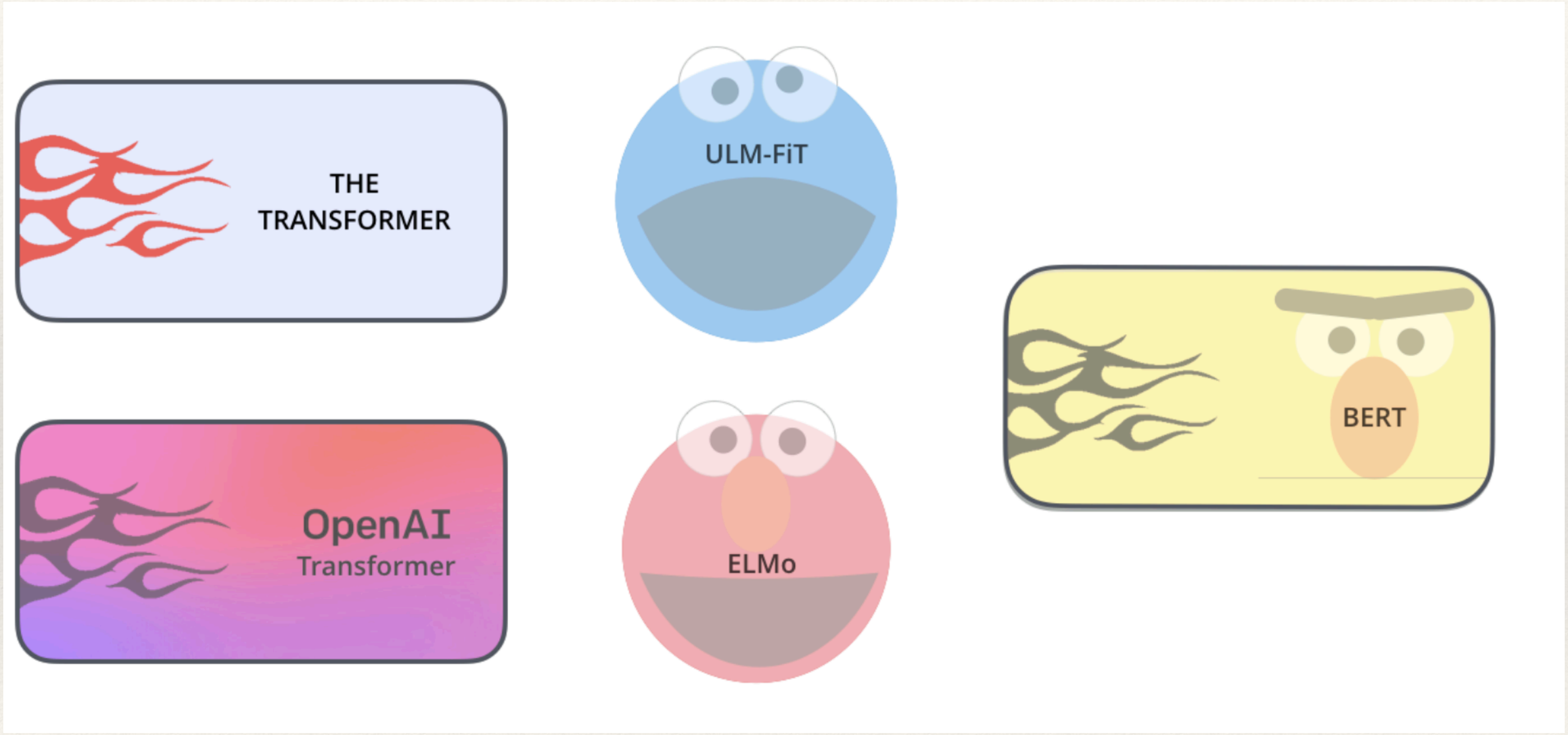
Transfer Learning in NLP

Navneet Kumar Chaudhary
Data Scientist
Aasaanjobs.com

Recent State of The Arts Models



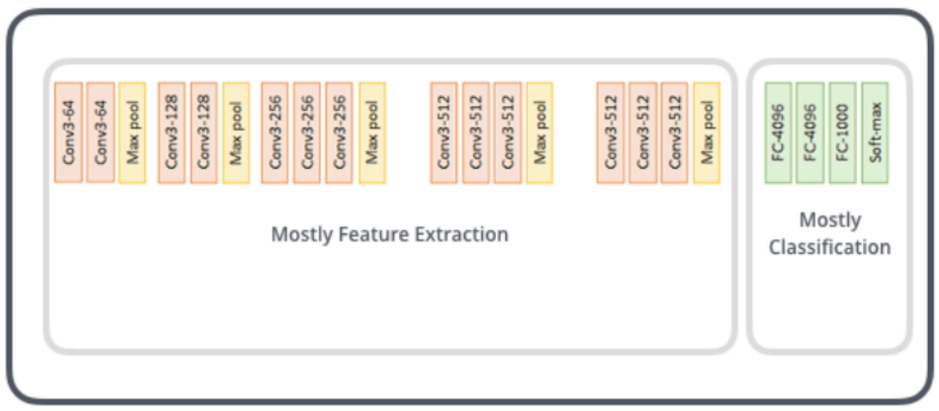
SOTA NLP Models

Image Sourced from https://jalammar.github.io/illustrated-bert/

Input

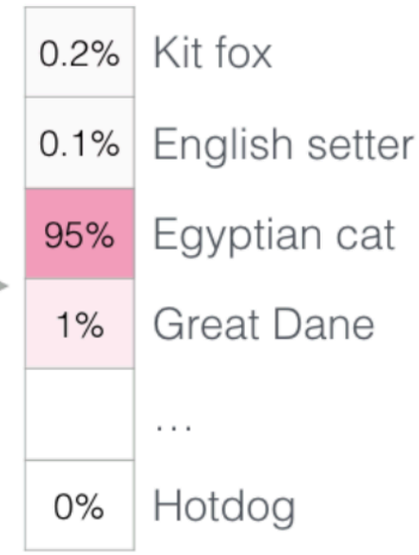
Features

VGG-16



Output

Prediction



Transfer Learning in CV and how we use embeddings

What is NLTK

- NLTK or The Natural Language ToolKit is a suite of libraries and programs for a variety of academic Text processing tasks:
- It has in built functionalities for Removing Stop words, Tokenization, Stemming, Lemmatizing

Stemming vs Lemmatization

Lemmatisation is closely related to **stemming**. The difference is that a stemmer operates on a single word without knowledge of the context, and therefore cannot discriminate between words which have different meanings depending on part of speech. However, stemmers are typically easier to implement and run faster, and the reduced accuracy may not matter for some applications.

For instance:

1. The word "better" has "good" as its lemma. This link is missed by stemming, as it requires a dictionary look-up.

2. The word "walk" is the base form for word "walking", and hence this is matched in both stemming and lemmatisation.

3. The word "meeting" can be either the base form of a noun or a form of a verb ("to meet") depending on the context, e.g., "in our last meeting" or "We are meeting again tomorrow". Unlike stemming, lemmatisation can in principle select the appropriate lemma depending on the context.

Word Embeddings Recap

- For words to be processed by machine learning models, they need some form of numeric representation that models can use in their calculation.
- Word2Vec showed that we can use a vector (a list of numbers) to properly represent words in a way that captures semantic or meaning-related relationships.
- Queen = King - Man + Woman
- Relationship between Country and their respective Capitals

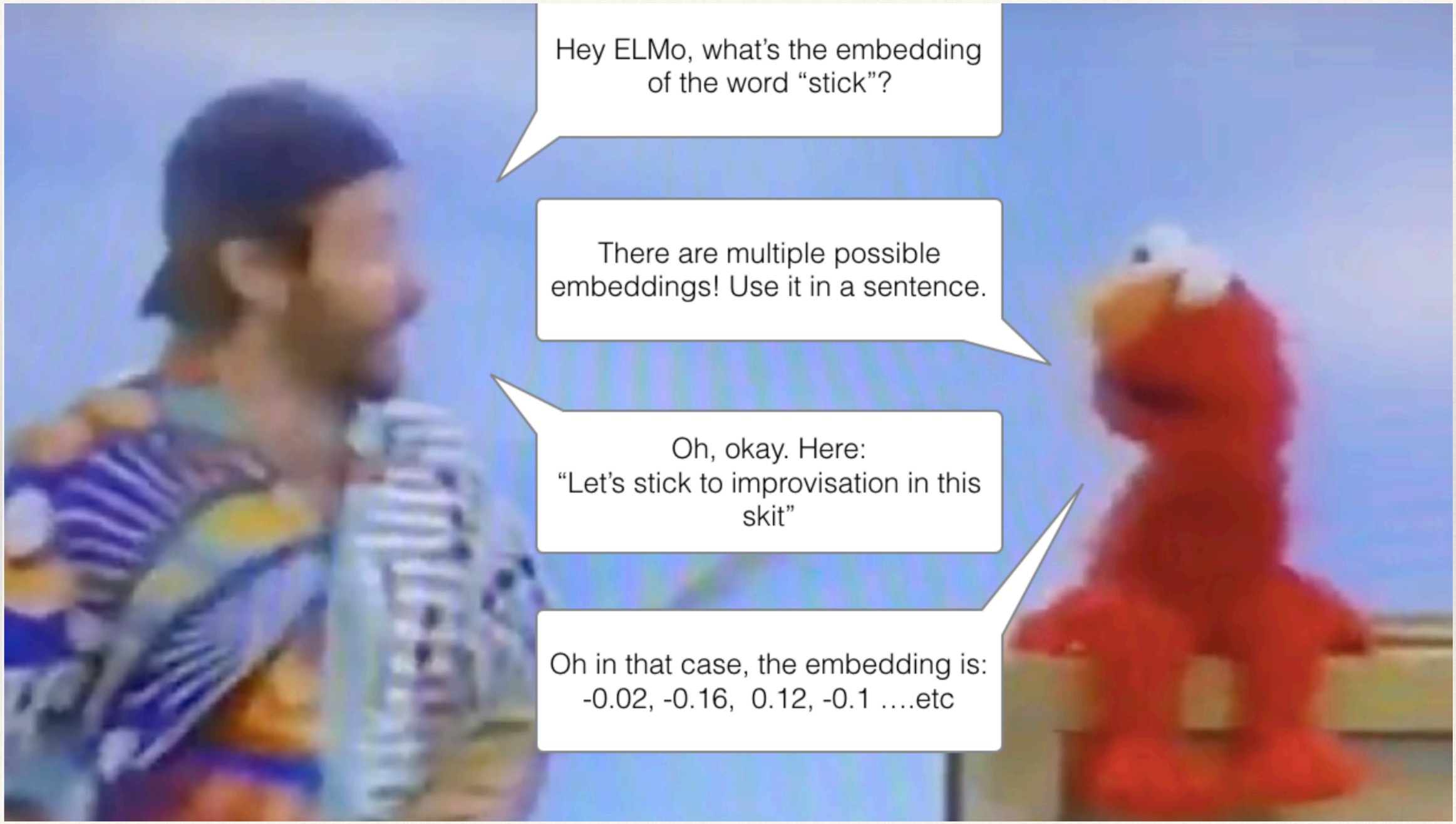
-0.34	-0.84	0.20	-0.26	-0.12	0.23	1.04	-0.16	0.31	0.06	0.30	0.33	-1.17	-0.30	0.03	0.09	0.35	-0.28	-(

The GloVe word embedding of the word "stick" - a vector of 200 floats (rounded to two decimals). It goes on for two hundred values.

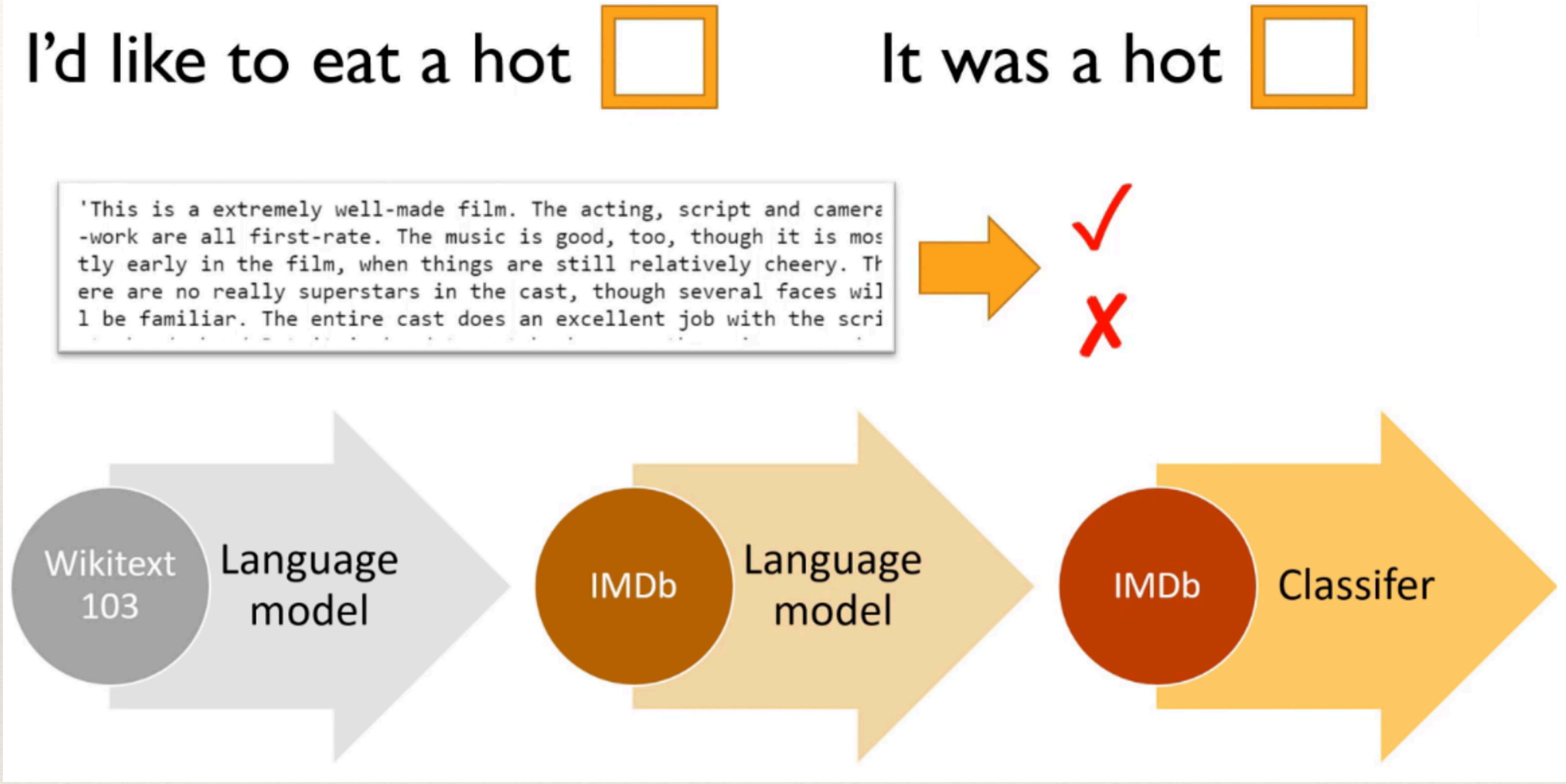
Limitations/Isuues in Word Embeddings

- Out of Vocabulary/Unknown words as we need to fix the vocabulary size(when a word is not known vector cannot be constructed deterministically)
- Cannot handle the shared representation of the same word. Meaning of a word depends on the context it is used.
- Our model won't be robust for new Languages, and thus we cannot use for incremental learning.

ELMO Context Matters



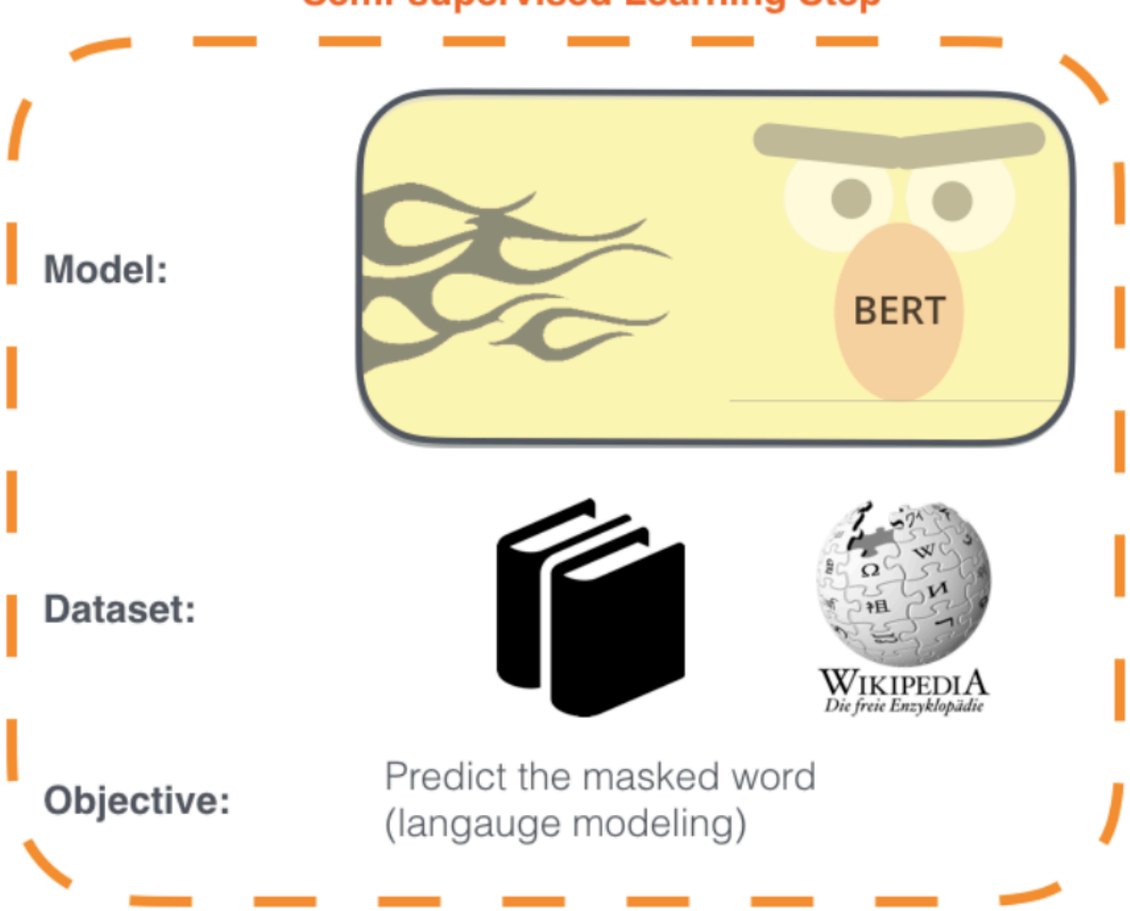
Context Aware Embeddings by ELMO



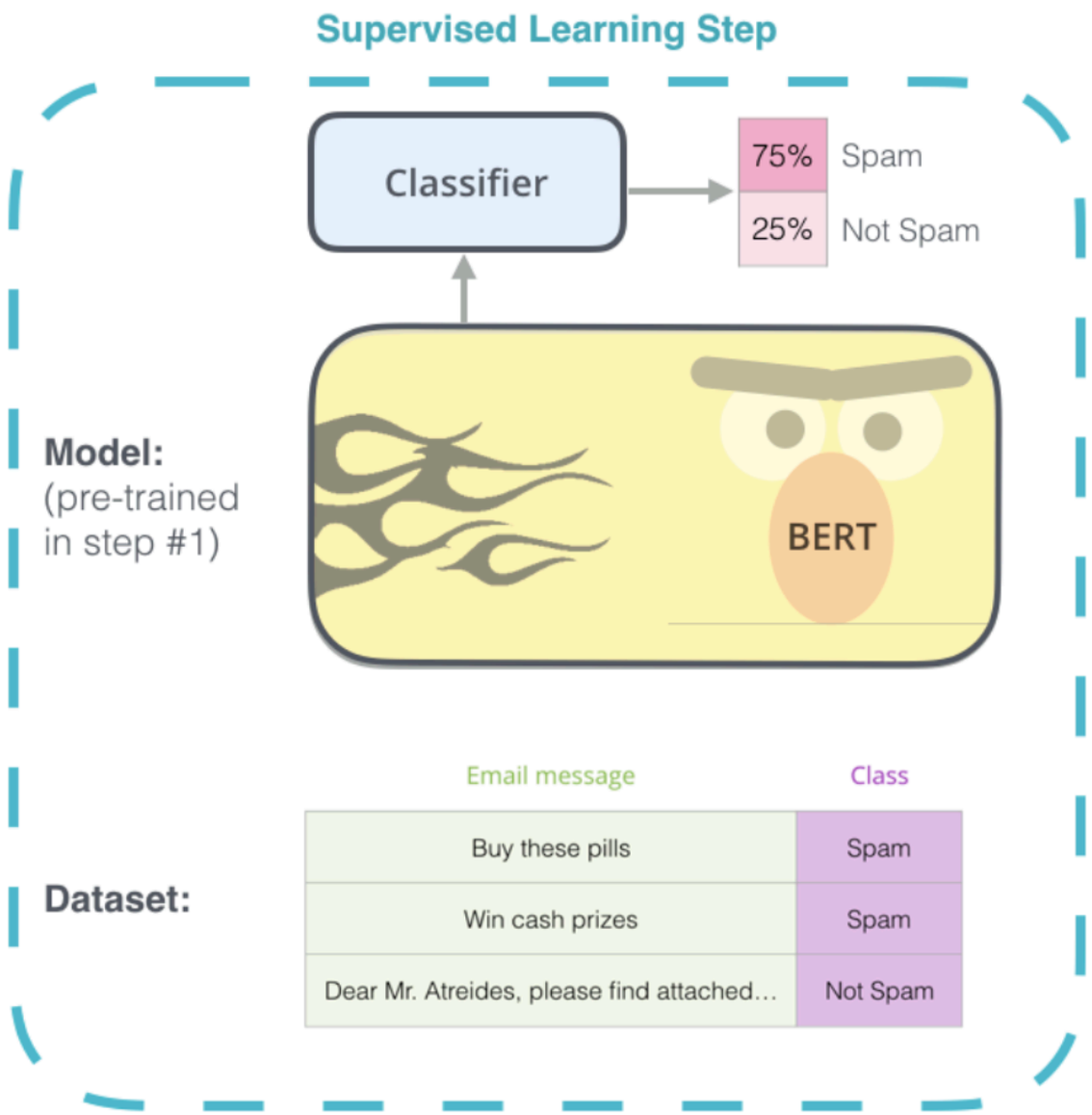
ULMFiT Approach to pre-training

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.

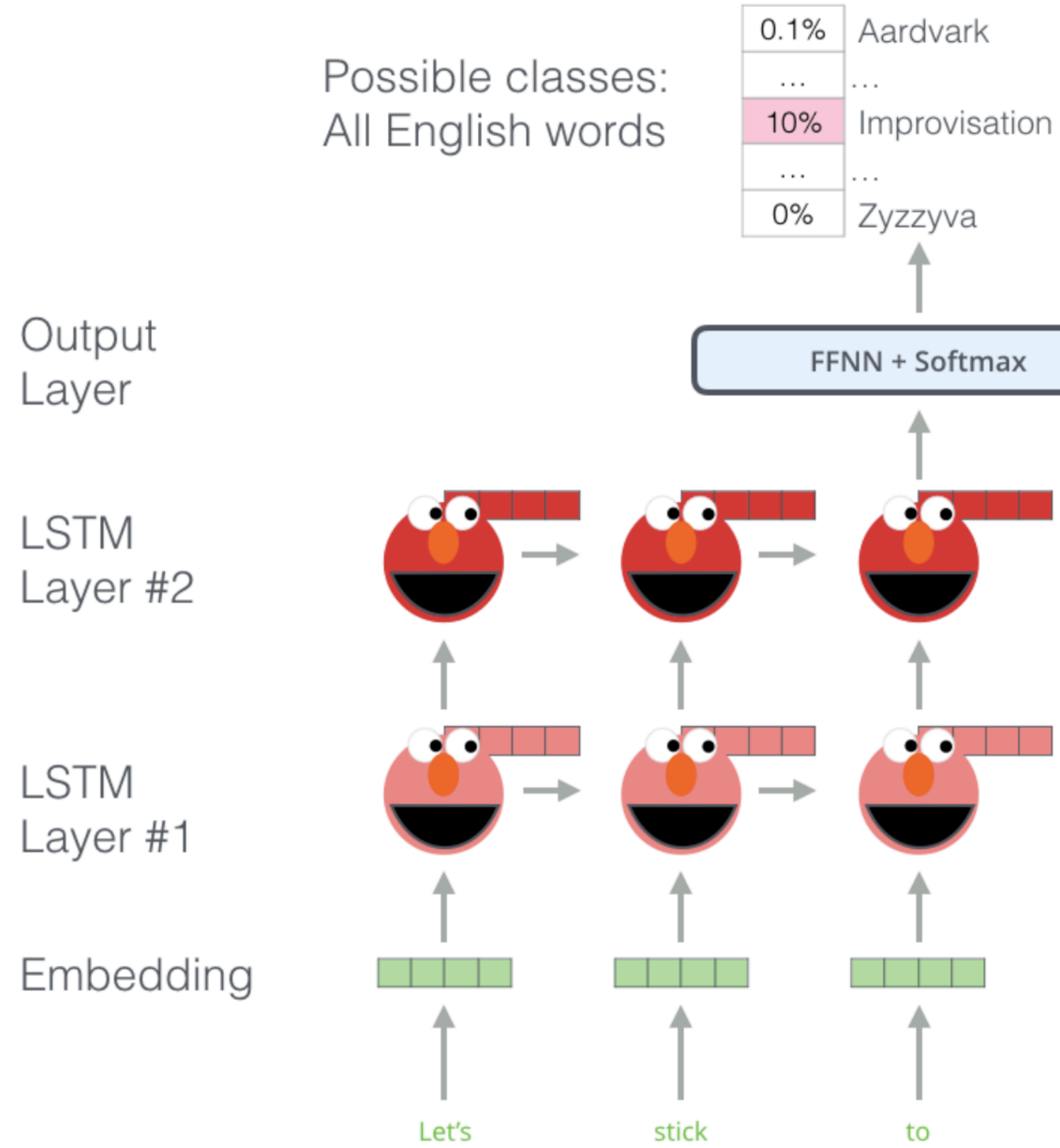


2 - Supervised training on a specific task with a labeled dataset.

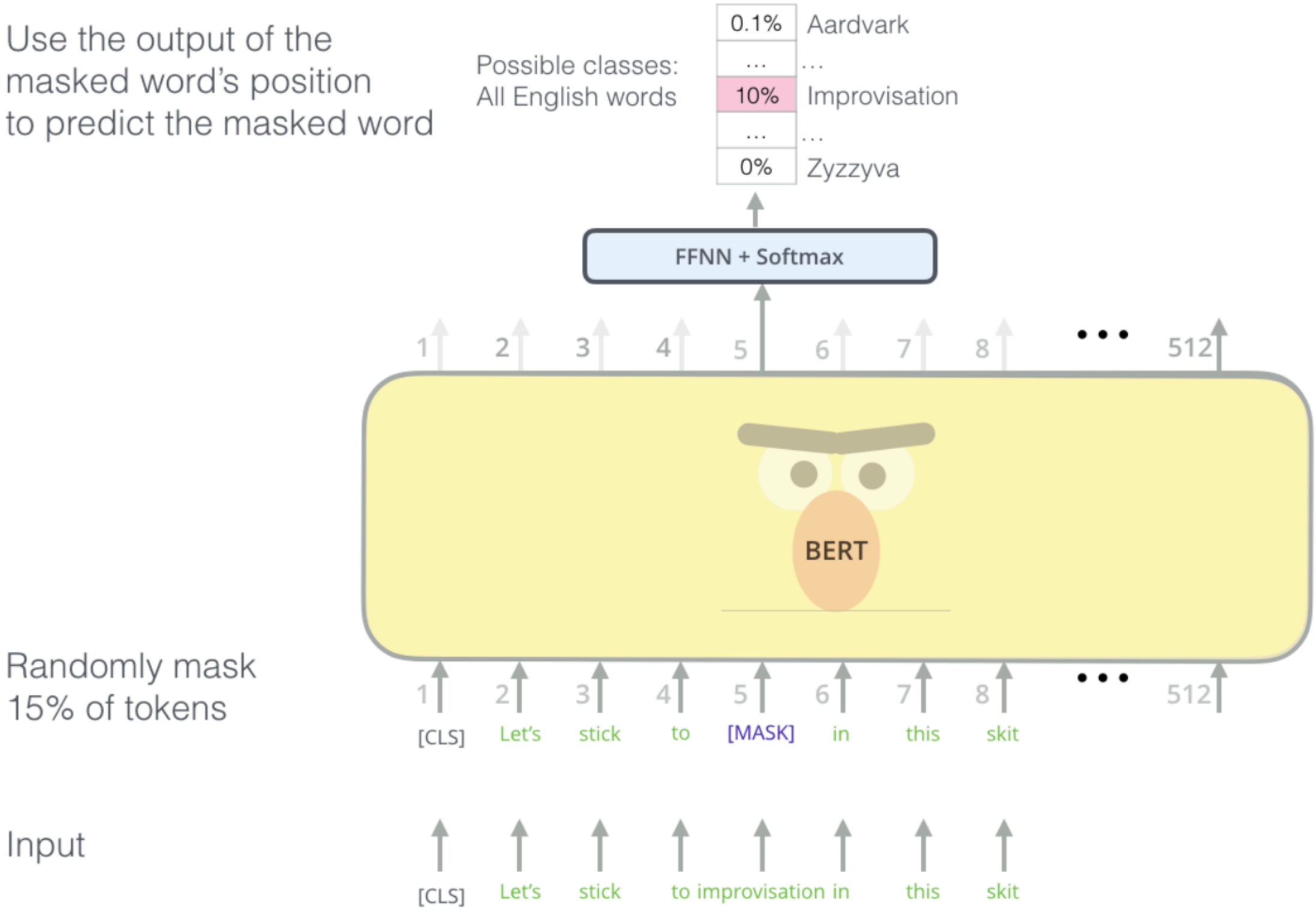


The two steps of how BERT is developed. You can download the model pre-trained in step 1 (trained on un-annotated data), and only worry about fine-tuning it for step 2. [Source for book icon].

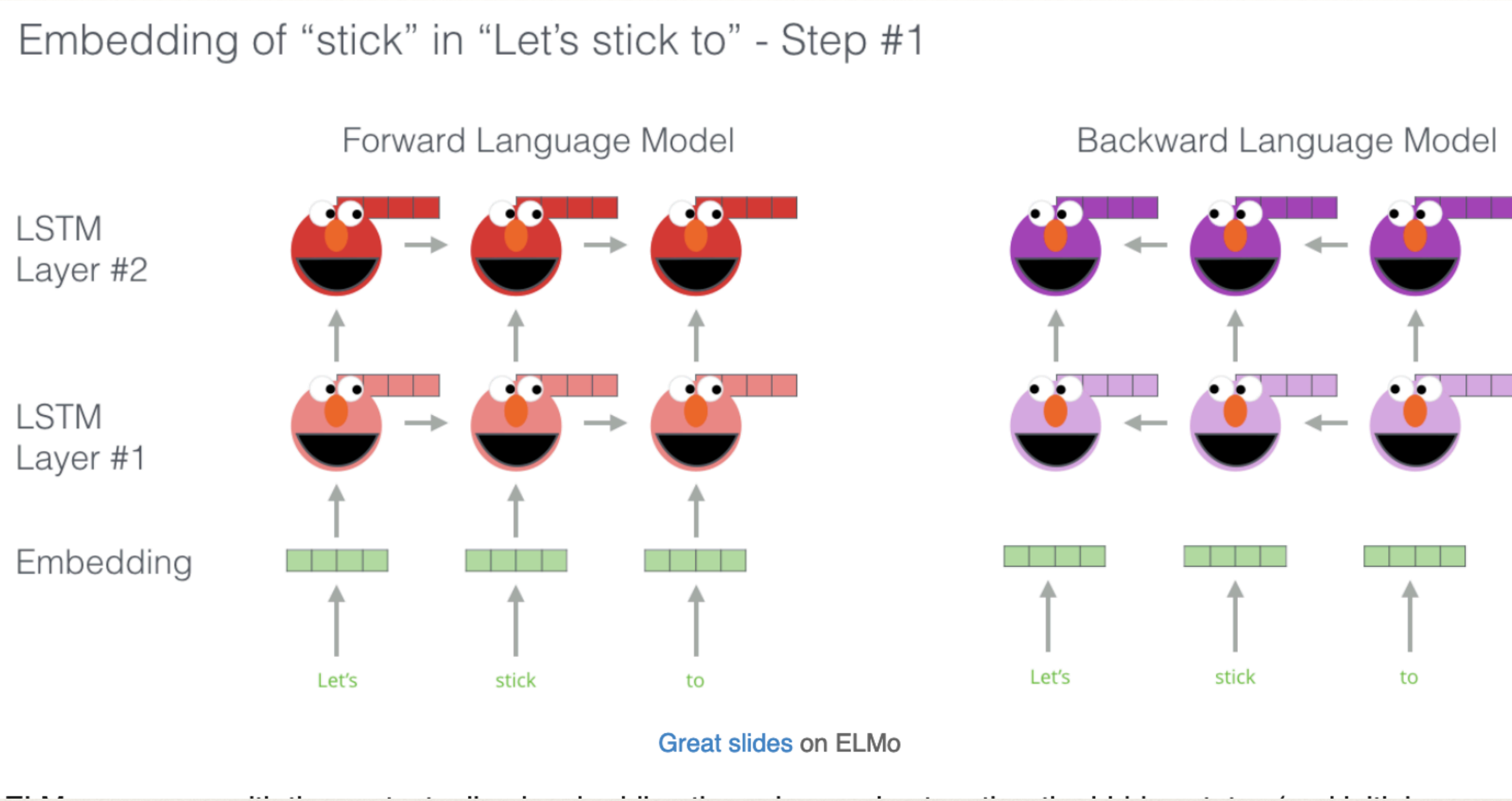
The idea for converting this to Transfer Learning



A step in the pre-training process of ELMo: Given "Let's stick to" as input, predict the next most likely word – a *language modeling* task. When trained on a large dataset, the model starts to pick up on language patterns. It's unlikely it'll accurately guess the next word in this example. More realistically, after a word such as "hang", it will assign a higher probability to a word like "out" (to spell "hang out") than to "camera".



BERT Pre-training Process

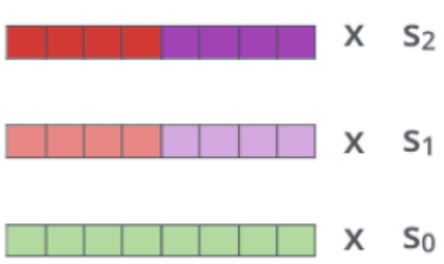


Step:1 Finding Context aware Embeddings

Embedding of “stick” in “Let’s stick to” - Step #2

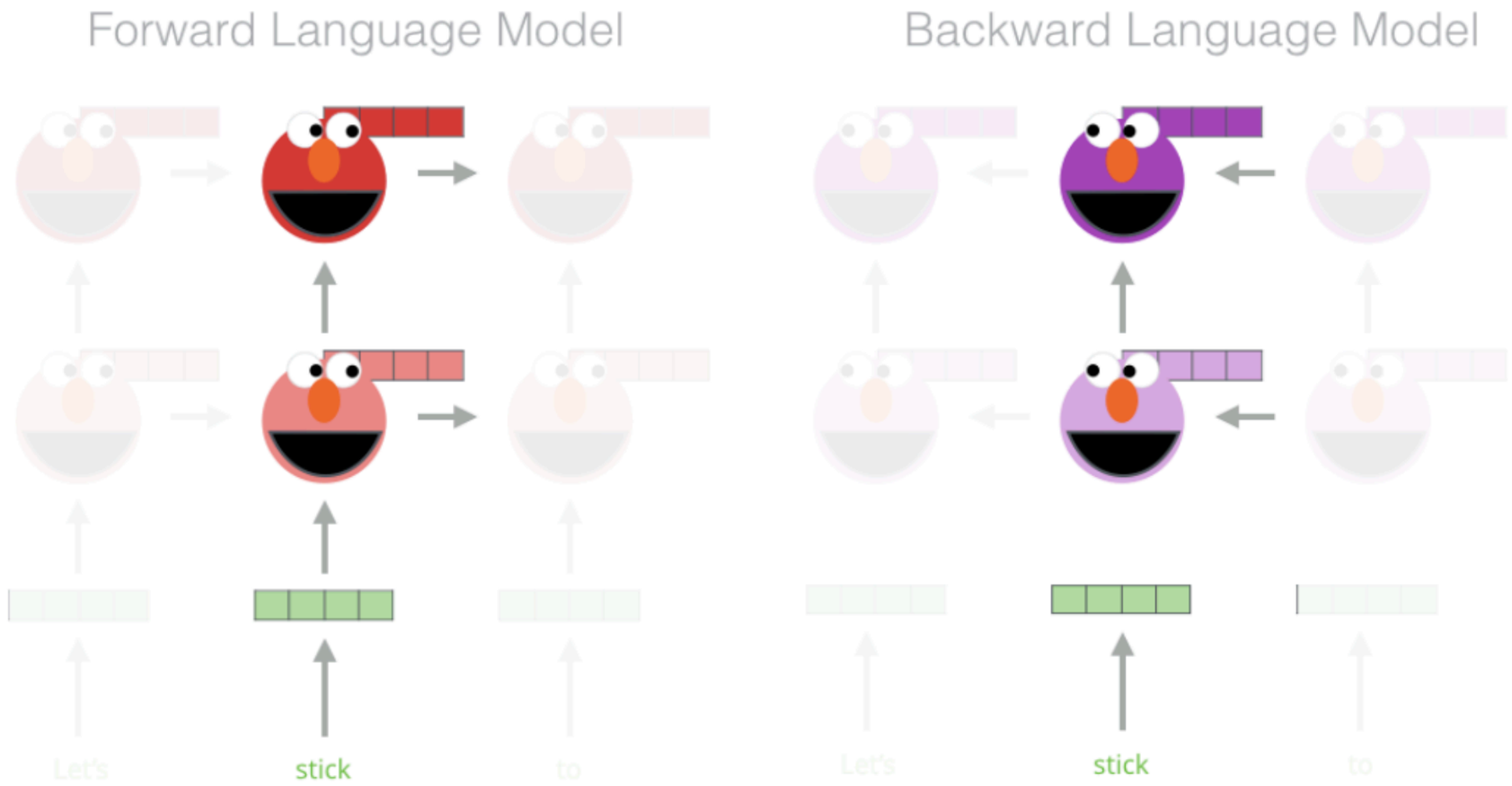
1- Concatenate hidden layers

2- Multiply each vector by a weight based on the task



3- Sum the (now weighted) vectors

ELMo embedding of “stick” for this task in this context



Step 2: Finding Context aware Embeddings

Why is ULMFiT Universal?

- Dataset independent. You start with wiki text LM and fine-tune for your dataset.
- Works across all documents and datasets of varying lengths.
- Architecture is consistent, same as we use ResNets for many CV tasks.
- Can work on very small datasets as well, as we already have a good LM to start with.

Classifier fine-tuning for Task Specific Weights

- Two additional linear blocks have been added. Each block uses batch normalization and a lower value of dropout
- ReLU is used as activation function in between the linear blocks.
- Softmax is used to provide the probability distribution over the target classes.
- Classifiers only take the embeddings provided by the LM and are always trained from scratch.

Results from ULMFiT

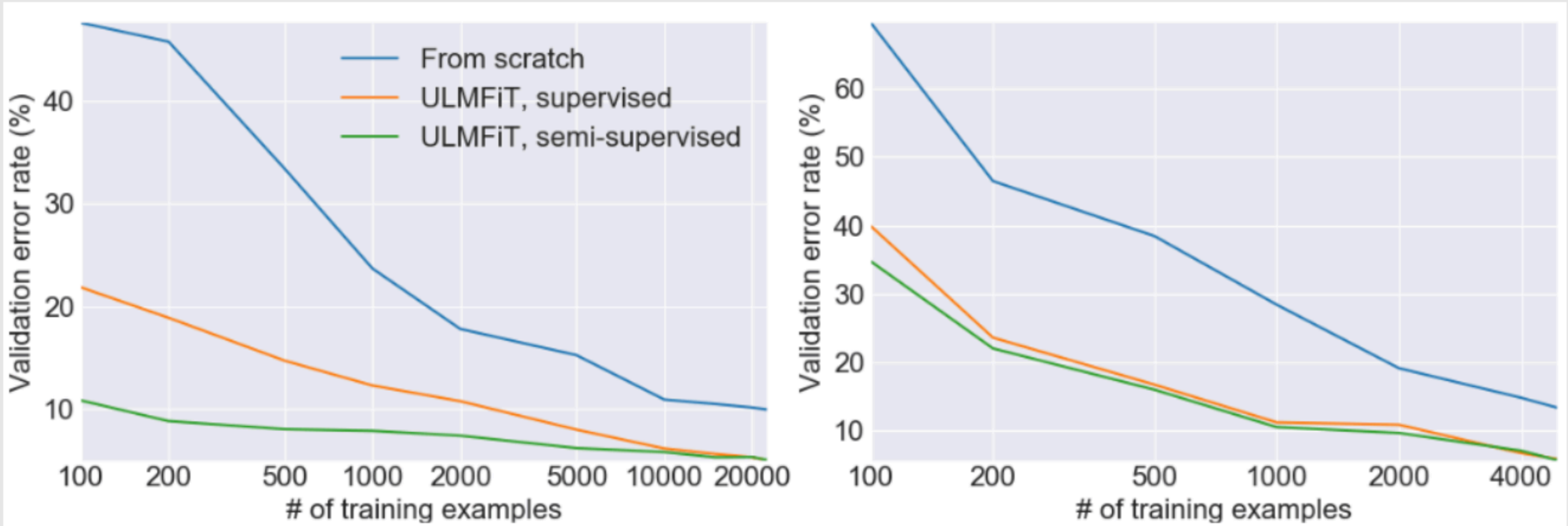


Figure 11. Validation error rate

Validation Error Rate ULMFiT vs Scratch

Acknowledgements

- "Images speak louder than words" and they were sourced from other blogposts and Google results.
- A lot of them are taken from this great blogpost by Jay Alammar https://jalammar.github.io/illustrated-bert/
- The results image is taken from the ULMFiT paper.

Thanks a Lot!!!

–Navneet Kumar Chaudhary