# Semantic search and similarity ranking

Ane Berasategi

18. July 2019

# Plan

## Part 1: Semantic search

- Introduction
- Ontology matching

## Part 2: Similarity ranking

- Word vectors
- Contextualized word embeddings
- ELMo
- BERT
- Flair

## Part 3: Experiments

- A
- A

## Part 4: Conclusion

# 1.1. Introduction

- **Lexical** search: literal matches of the query words.
  - Anthony Hopkins age → good
  - How old is Anthony Hopkins? → bad
- **Semantic** search: search with meaning, understand the query and the intention of the user
  - Why is my laptop overheating?
  - Why do bees follow me?
  - How many continents are there in the world?
    - Correct answer: 6

# 1.2. Ontology matching

The search engine has a huge knowledge graph / ontology with past searches, a representation of semantic relations between documents.

Pipeline:

1. New query arrives
2. Query broken into root terms: POS tagging removal, NER, error correction, conversion to embeddings, etc
3. Return the closest/more relevant/semantically most similar documents from the ontology (similarity ranking)

# Plan

## Part 1: Semantic search

- Introduction
- Ontology matching

## Part 2: Similarity ranking

- Word vectors
- Contextualized word embeddings
- ELMo
- BERT
- Flair

## Part 3: Experiments

- A
- A

## Part 4: Conclusion

# 2.1. Word vectors: history

*"A word is characterized by the company it keeps"* – Firth, 1958

How to quantify and categorize semantic similarities between linguistic items based on their distributional properties?

- 1980s, LSA: reduce the number of dimensions using singular value decomposition
- 2000, Bengio et al.: reduce the high dimensionality of words representations in contexts by learning a distributed representation for words
- 2013, Mikolov et al., word2vec: word embedding toolkit to train word vectors in NNs, faster than n-gram models
- Pre-trained word embeddings became the norm (word2vec, GloVe, FastText) as input to NNs

# 2.2. Contextualized word embeddings

- Word embeddings: each word gets an embedding vector
- Irrelevant of the context, part of speech, polysemy

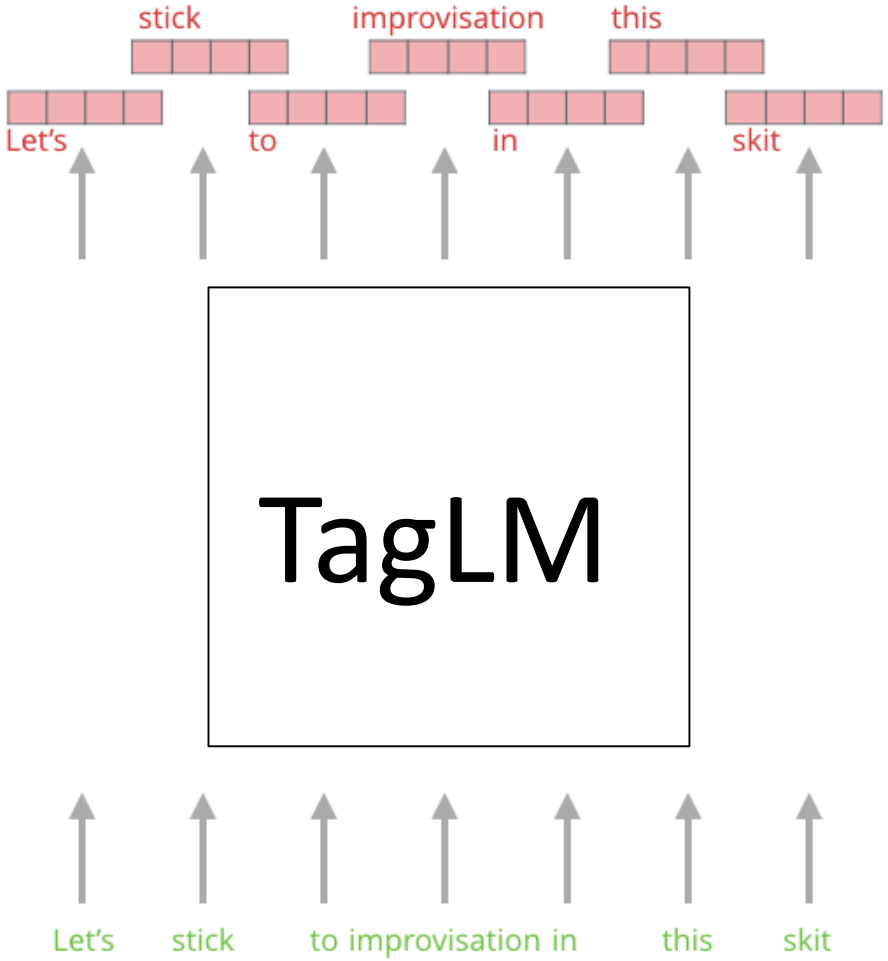
- 2017, Peters et al.: **TagLM**: give the words an embedding vector based on its context, in order to:
  - capture word meaning in that context
  - capture other contextual information

Semi-supervised sequence tagging with bidirectional language models, Peters et al., 2017

# 2.2. Contextualized word embeddings

- Word embeddings: each word gets an embedding vector
- Irrelevant of the context, part of speech, polysemy

- 2017, Peters et al.: **TagLM**: give the words an embedding vector based on its context, in order to:
  - capture word meaning in that context
  - capture other contextual information




Semi-supervised sequence tagging with bidirectional language models, Peters et al., 2017

# 2.3. TagLM (April 2017)

Formally described as: **semi-supervised** approach to add contextual embeddings to word embeddings from **bidirectional language models**

- **Language model** (LM): computes the probability of a token given a token sequence.
- Uses LSTM architecture, produces LM embeddings
  - Forward LM: Given the previous token sequence in a sentence, predict next token
  - Backward LM: Given the future token sequence, predict previous token
  - Bidirectional LM: forwLM and backLM trained separatedly and then concatenated to form the biLM embeddings
- **Semi-supervised** approach:
  - the biLM is trained on a large unlabelled corpus
  - the biLM embeddings are added as additional input to the NLP task

Semi-supervised sequence tagging with bidirectional language models, Peters et al., 2017

# 2.3. **TagLM:** paper remarks

- Applied to sequence labelling tasks: assigning a categorical label to each member of a sequence of words.
- Using both forwLM and backLM embeddings boosts performance over forwLM embeddings
- Transfer learning: the biLM embeddings trained in one domain can be transferred to another
- Inputs to **TagLM**:
  - Character representation model, CNN or RNN
  - Token embeddings, initialized using pre-trained word embeddings
  - Recurrent LM: LSTM model with multiple layers
- Output from **TagLM**: a single context-independent representation for each word, **the output layer of the LSTM**.

Semi-supervised sequence tagging with bidirectional language models, Peters et al., 2017

# 2.3. ELMo (February 2018)

- Why just the last layer of the LSTM? Use all layers

Deep contextualized word representations, Peters et al., 2018

# 2.3. ELMo (February 2018)

- Why just the last layer of the LSTM? Use all layers
- Deep “bidirectional” contextual embeddings (BERT paper disagrees). ForwLM and backLM share some weights, not completely independent (unlike paper #1), afterwards shallow concatenation

Deep contextualized word representations, Peters et al., 2018

# 2.3. ELMo (February 2018)

- Why just the last layer of the LSTM? Use all layers
- Deep “bidirectional” contextual embeddings (BERT paper disagrees). ForwLM and backLM share some weights, not completely independent (unlike paper #1), afterwards shallow concatenation
- The embeddings are functions of the entire input sentence, computed on top of a 2 layer biLM with character convolutions, as a linear function of the internal network states.

Deep contextualized word representations, Peters et al., 2018

# 2.3. ELMo (February 2018)

- Why just the last layer of the LSTM? Use all layers
- Deep “bidirectional” contextual embeddings (BERT paper disagrees). ForwLM and backLM share some weights, not completely independent (unlike paper #1), afterwards shallow concatenation
- The embeddings are functions of the entire input sentence, computed on top of a 2 layer biLM with character convolutions, as a linear function of the internal network states.
- Higher-level layers of the LSTM capture context-dependent aspects of word meaning
- Lower-level layers of the LSTM capture aspects of syntax and can be used for POS tagging

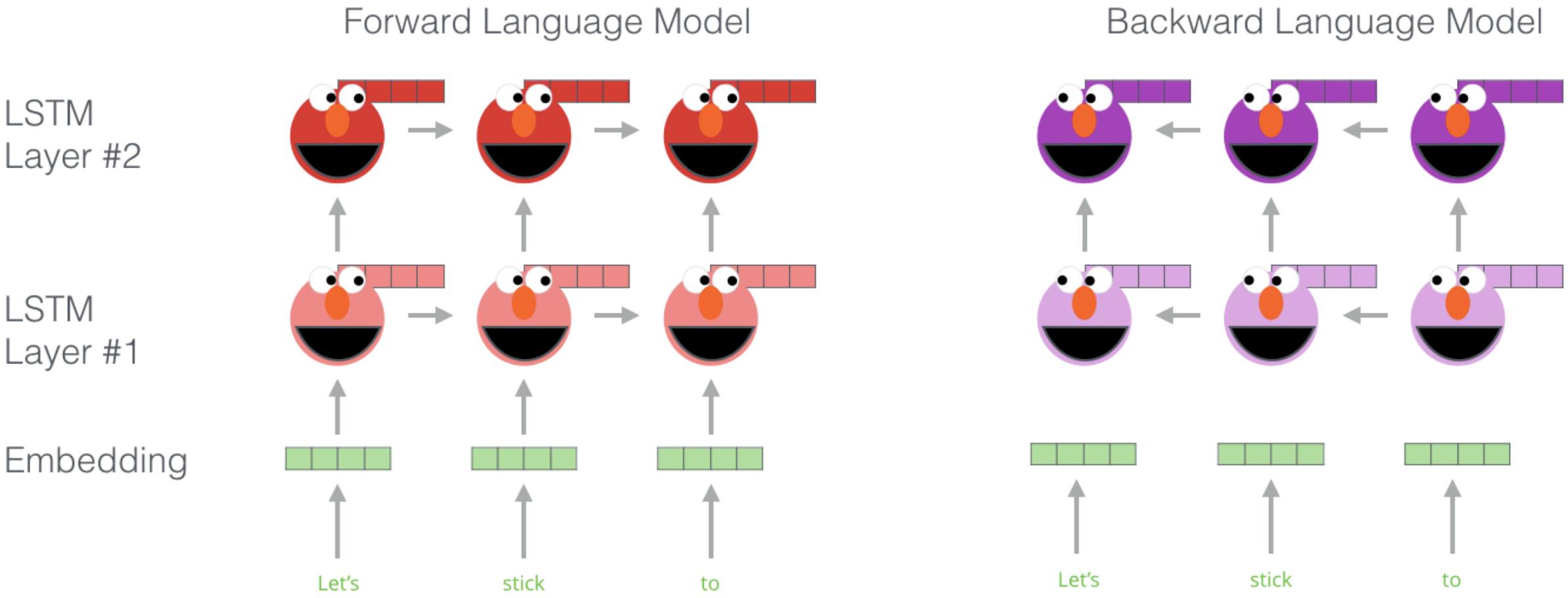
Deep contextualized word representations, Peters et al., 2018

# 2.3. ELMo (February 2018)

- Why just the last layer of the LSTM? Use all layers
- Deep “bidirectional” contextual embeddings (BERT paper disagrees). ForwLM and backLM share some weights, not completely independent (unlike paper #1), afterwards shallow concatenation
- The embeddings are functions of the entire input sentence, computed on top of a 2 layer biLM with character convolutions, as a linear function of the internal network states.
- Higher-level layers of the LSTM capture context-dependent aspects of word meaning
- Lower-level layers of the LSTM capture aspects of syntax and can be used for POS tagging
- ELMo is a **feature-based approach** for contextual embeddings: task-specific architectures that include the pre-trained representations as additional features
  - different architectures for different NLP tasks
  - The embeddings are added as additional inputs to the NLP task

Deep contextualized word representations, Peters et al., 2018

# Embedding of ‘stick’ in ‘Let’s stick to’: step #1




[The Illustrated BERT, ELMo, and co.,](#) Jay Alammar, 2018

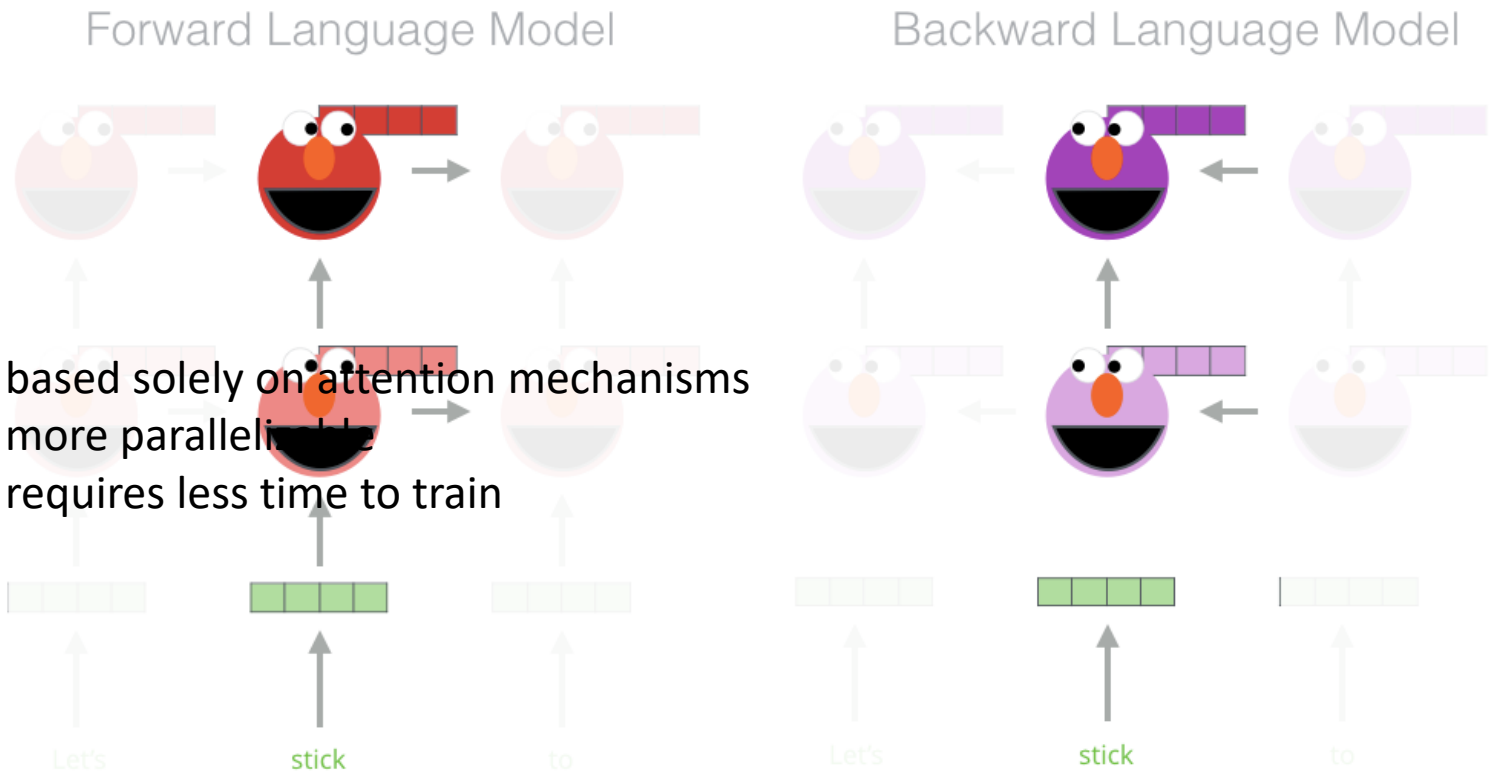
# Embedding of ‘stick’ in ‘Let’s stick to’: step #2

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




based solely on attention mechanisms
more parallelizable
requires less time to train

[The Illustrated BERT, ELMo, and co.,](#) Jay Alammar, 2018

# 2.4. The transformer (june 2017) vs LSTM

- Recurrent, sequential models can't parallelize, which become critical at longer sequence lengths
- The transformer:
  - No sequence, based solely on attention mechanisms
  - Deals with long-term dependencies better than LSTMs
  - More parallelizable
  - Requires less time to train
  - first transduction model relying entirely on **self-attention** to compute representations of its input and output without using sequence-aligned RNNs or convolutions


Attention Is All You Need, Vaswani et al., 2018
The illustrated Transformer, Jay Alammar, 2018

# 2.4. BERT

Expectation

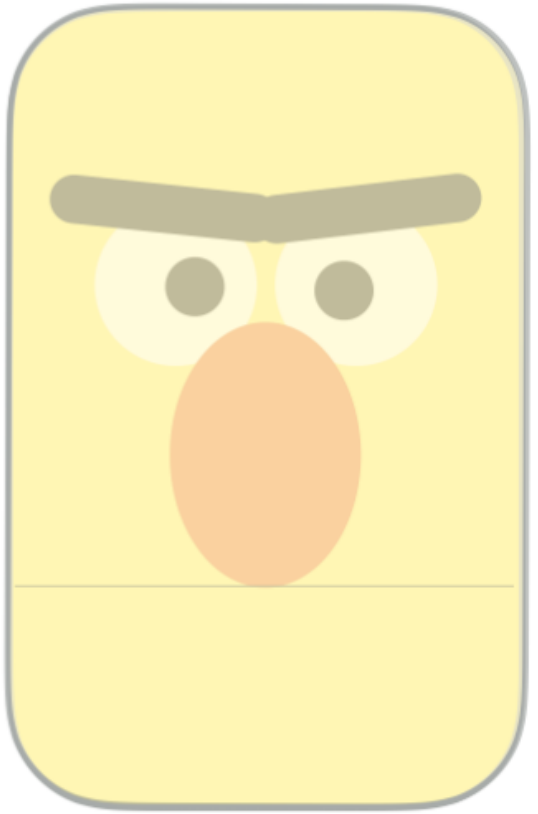
# 2.4. BERT

Expectation

Reality

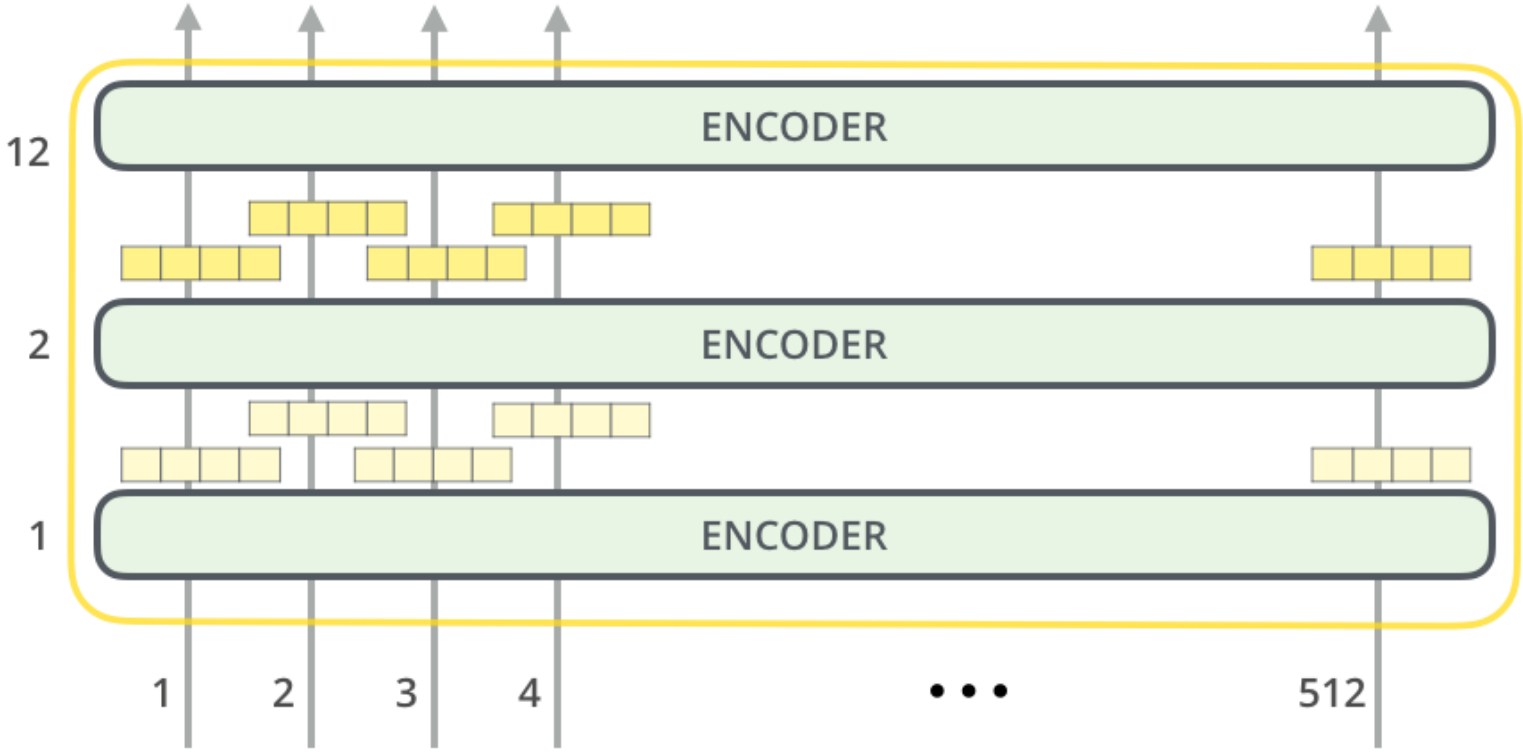
# 2.4. BERT: Bidirectional Embedding Representations from Transformers (October 2018)

- BERT pretrains deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. (unlike ELMo)
  - ELMo uses shallow concatenation of independently trained forward and backward LMs
- BERT is fine-tuned: minimal task-specific parameters. trained on NLP tasks by simply fine-tuning **all** pre-trained parameters end-to-end. Compared to pre-training, fine-tuning is relatively inexpensive
- BERT uses a masked LM, randomly masking some tokens in the sentence and training to predict the masked token. MLM enables the representation to fuse the left and right context
- **Pre-trained BERT can be used to create contextualized word embeddings**



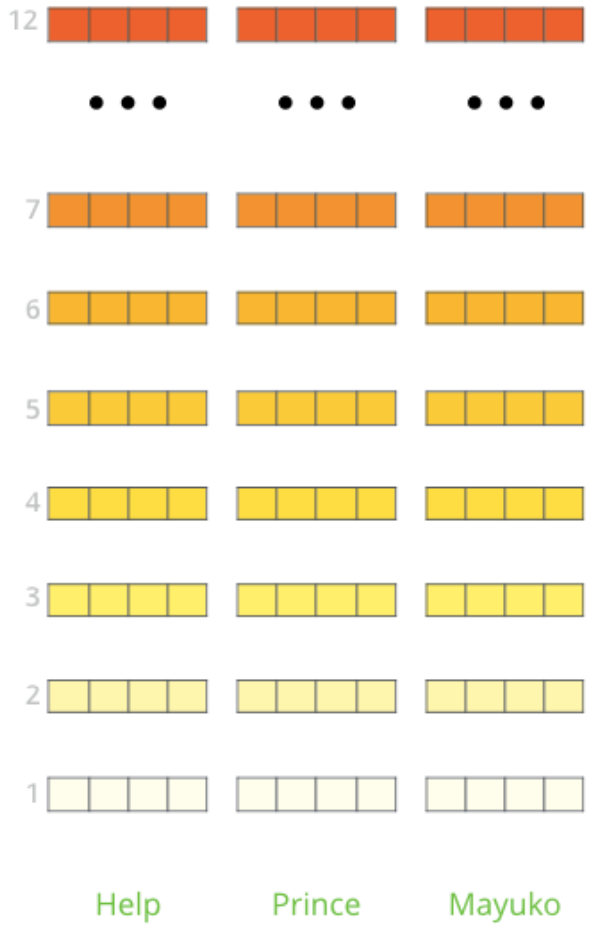
BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, Devlin et al., 2018
The Illustrated BERT, ELMo, and co., Jay Alammar, 2018

## Generate Contexualized Embeddings



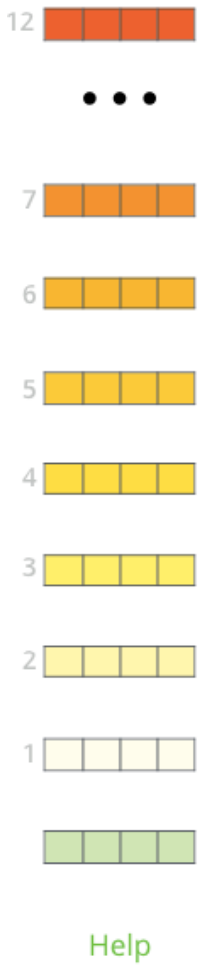

BERT

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?**

**What is the best contextualized embedding for "Help" in that context?**
For named-entity recognition task CoNLL-2003 NER




| | | Dev F1 Score |
|---|---|---|
| First Layer | Embedding | 91.0 |
| Last Hidden Layer | 12 | 94.9 |
| Sum All 12 Layers | 12 + ... + 2 + 1 = | 95.5 |
| Second-to-Last Hidden Layer | 11 | 95.6 |
| Sum Last Four Hidden | 12 + 11 + 10 + 9 = | 95.9 |
| Concat Last Four Hidden | 9 10 11 12 | 96.1 |

The layer(s) to use depend on the NLP task

Some tasks look at similarities between sentences, others care more about syntax and POS tagging

# 2.5. flair

- (didn’t read paper yet)

Contextual String Embeddings for Sequence Labeling, Akbik et al., 2018

# Plan

## Part 1: Semantic search

- Introduction
- Ontology matching

## Part 2: Similarity ranking

- Word vectors
- Contextualized word embeddings
- ELMo
- BERT
- Flair

## Part 3: Experiments

- A
- A

## Part 4: Conclusion

# 3.1. Experiments

- *Awesome explanation about the results*

# 3.1. Question: does word order matter?

- The ‘opposite’ sentence (B invites A instead of A invites B) has a very high similarity (0.96-0.99 in English, 0.983-0.99 in basque)
  - The initial intuition was that the opposite sentence wouldn’t be ranked to high because it has the ‘opposite’ meaning but the two sentences belong to the same semantic space → semantically similar
- Changing a proper noun by lunch decreases similarity to 0.86-0.97 (en), 0.93-0.98 (eu)
  - which is logical
- A random ordering of words yields a similarity of 0.78-0.93 (en), 0.77-0.97 (eu)
  - still higher than expected
- Sentences in basque have a higher similarity overall
- Yes, word order matters

# 3.2. What is the impact of lexical similarity?

- The ‘negation’ of the sentence (A didn’t invite B) also has a high semantic similarity
  - Same intuition as before, but since they belong to the same semantic space → semantically similar
- “the doctor told the patient he was a fraud” has a smaller similarity range in Basque (0.65-0.92) than in English (0.84-0.93)
- “that is a matter between the doctor and the patient”:
  - English: lowest in ELMo (0.63), gradual in BERT uncased, flair and highest in BERT cased (0.89)
  - Basque: very low similarities in ELMo and flair (0.45-0.55), very high in the 2 BERTs (0.82-0.85)
- “the child and the grandfather got invited for lunch” has very high similarities in BERT cased (0.91 eu, 0.95 en)
- The compound “invite for lunch” has more weight than the subject and object

# 3.3. What is the impact of synonyms?

# 3.4. What is the impact of out of vocab words?

# Plan

## Part 1: Semantic search

- Introduction
- Ontology matching

## Part 2: Similarity ranking

- Word vectors
- Contextualized word embeddings
- ELMo
- BERT
- Flair

## Part 3: Experiments

- A
- A

## Part 4: Conclusion

# 4. Conclusion

- *Awesome conclusion*

# Semantic search and similarity ranking

Ane Berasategi

18. July 2019