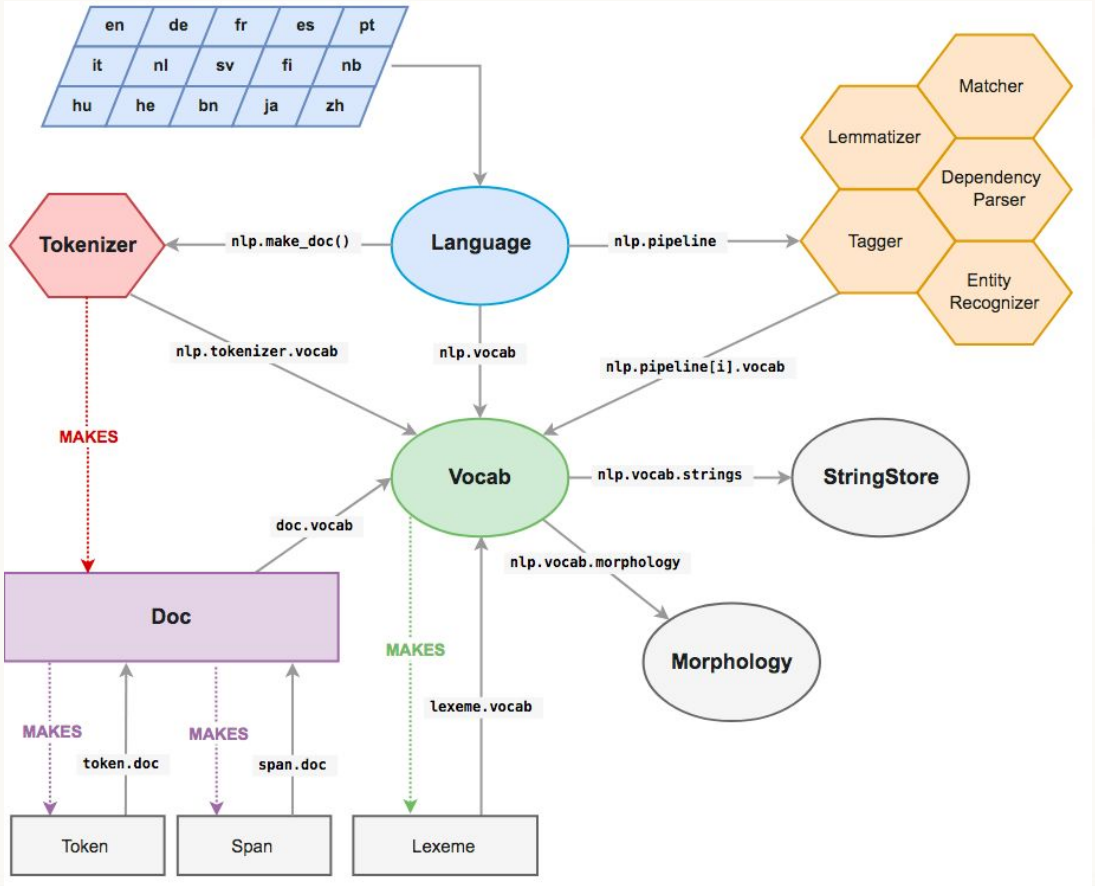
Global Big Data Conference

www.globalbigdataconference.com

Twitter : @bigdataconf

#GBDC

NLP preprocessing

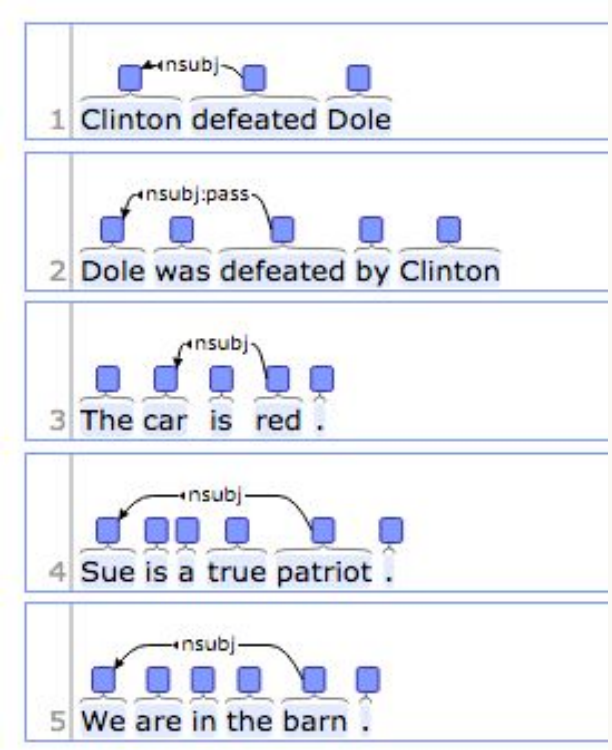


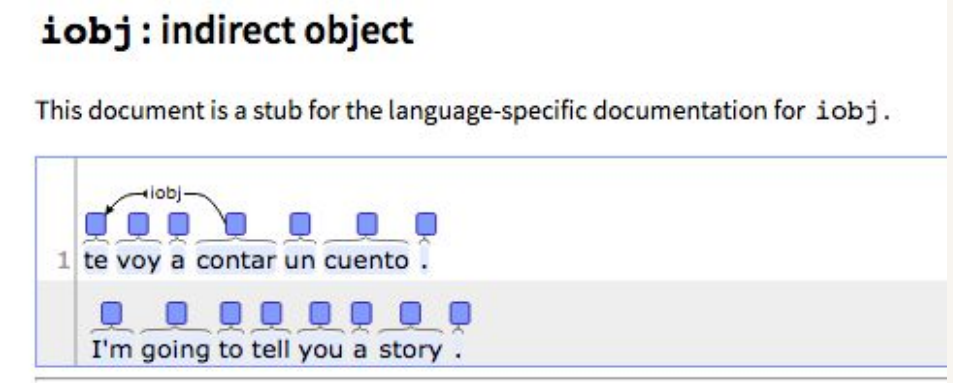
Preprocessing example

```
doc=nlp("this is one of the examples of tokenization. U.K won in the World Cup yesterday.")
for token in doc:
    print(token.text, token.pos_, token.lemma_, token.dep_)
```

```
this DET this nsubj
is VERB be ROOT
one NUM one attr
of ADP of prep
the DET the det
examples NOUN example pobj
of ADP of prep
tokenization NOUN tokenization pobj
. PUNCT . punct
U.K PROPN u.k nsubj
won VERB win ROOT
in ADP in prep
the DET the det
World PROPN world compound
Cup PROPN cup pobj
yesterday NOUN yesterday npadvmod
. PUNCT . punct
```

Dependency parsing- example





Dependency parsing- example

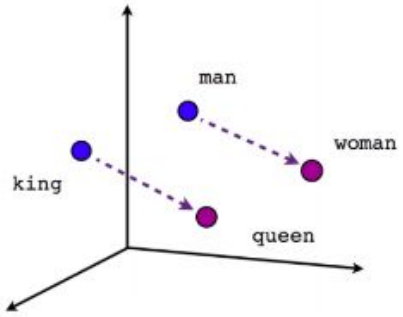
conj : conjunct

A conjunct is the relation between two elements connected by a coordinating conjunction, such as *and, or,* etc. We treat conjunctions asymmetrically: The head of the relation is the first conjunct and all the other conjuncts depend on it via the `conj` relation.

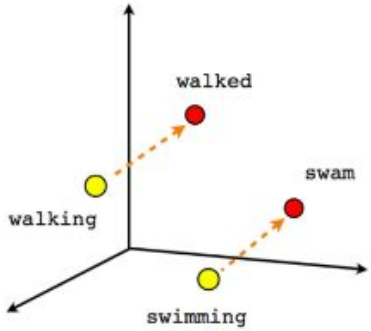
Coordinated clauses are treated the same way as coordination of other constituent types:

Word embeddings

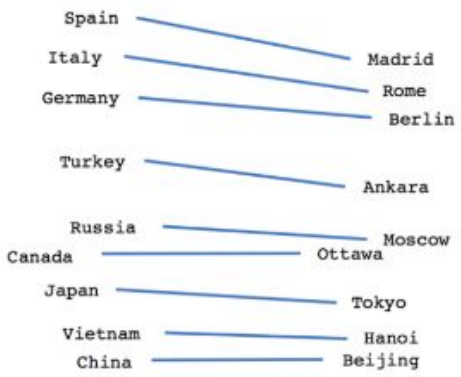
-Vectorial Representation of language
-Solve data sparsity of bag of words models capturing semantic relationships
(eg: dogs and cats are animals)
-Semantically similar words are mapped to nearby points (embedded next to each other)



Male-Female



Verb tense



Country-Capital

How are word embeddings computed*

-Maximum likelihood. Maximize probability of word given a context

$$J_{\text{ML}} = \log P(w_t|h) = \text{score}(w_t, h) - \log\left(\sum_{\text{Word w' in Vocab}} \exp\{\text{score}(w', h)\}\right).$$

- Negative sampling. Use just a limited set k of noise words

$$J_{\text{NEG}} = \log Q_\theta(D=1|w_t, h) + k \mathop{\mathbb{E}}_{\tilde{w}\sim P_{\text{noise}}} [\log Q_\theta(D=0|\tilde{w}, h)]$$

*Extracted from tensorflow tutorial
https://www.tensorflow.org/tutorials/representation/word2vec

How are word embeddings computed*

-Update embedding parameters to maximize the objective function

-Derive gradient of loss with respect to embedding parameters

-Update embeddings taking a small step in direction of gradient

-Repeating process over the entire training set, the embedding vectors for each word will be moved until the model is successful at discriminating real words from noise words (ie objective function maximized)

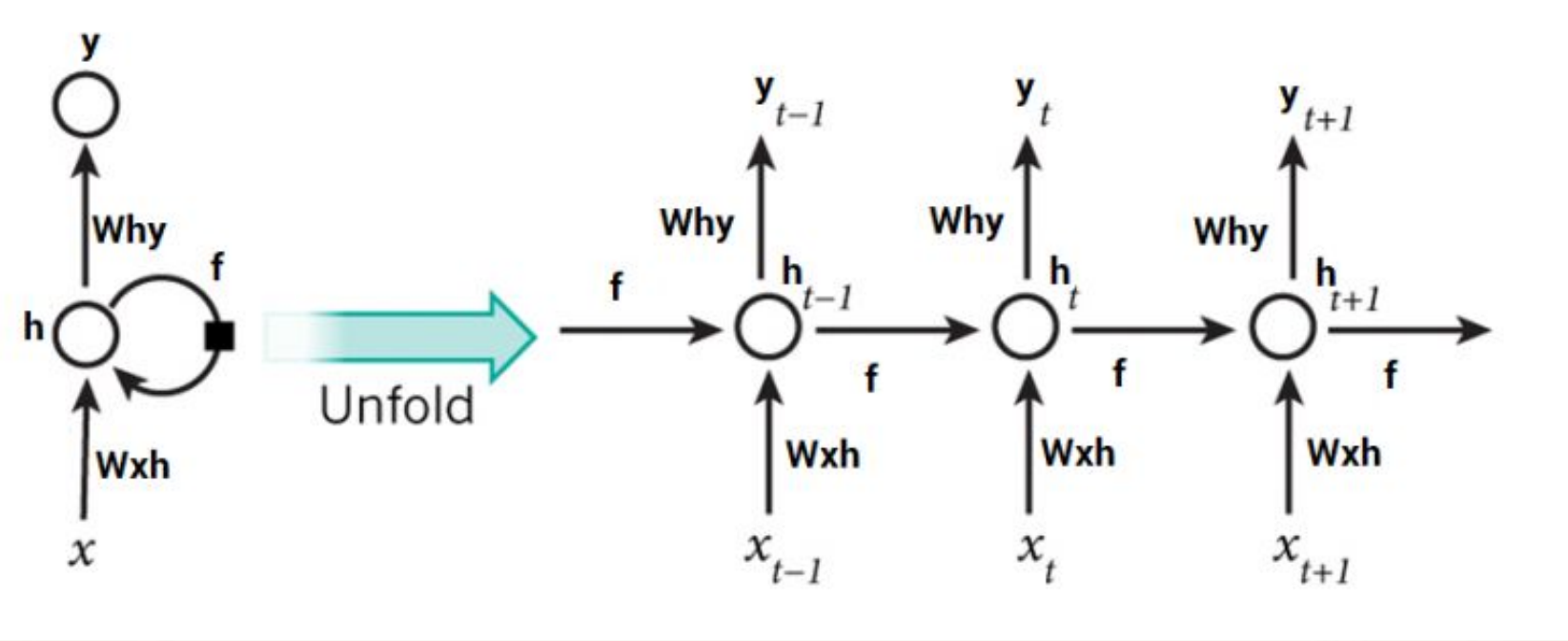
```
the quick brown fox jumped over the lazy dog
```

```
(quick, the), (quick, brown), (brown, quick), (brown, fox), ...
```

$$J_{\text{NEG}}^{(t)} = \log Q_\theta(D = 1|\text{the, quick}) + \log(Q_\theta(D = 0|\text{sheep, quick}))$$

RNN (recurrent neural networks)

-Take into account previous ocurrences/context apart from the input.
-Challenge? Only short term memory



LSTM (long short term memory)

Input (xt) + previous cell output (ht-1)

-Forget gate. What to forget from previous memory state (old memory)
-New memory/input gate: How much newly computed state you want to let through
-Output gate: How much of the internal state to expose to the network (next stage and NN)