Language Representation

- Context Free Embeddings

- Context Based Embeddings

By: Kanav Bansal
(That AI Guy)

Language Representation

By: Kanav Bansal
(That AI Guy)

Various Text Representation Techniques:

1. Bag of words → Captures frequency
2. TFIDF → Captures Word Importance
3. W2V → Captures Semantic Meaning by learning the relationships b/w different words. → This is done in a way that words which are similar will have vector representation closer to each other.

text → $\boxed{\text{Algorithm}}$ → W2v Model

↑
Powerful Architecture

Ques: What do you mean by Semantic Meaning?
Ans: Semantic meaning is about understanding a deeper meaning & relationships b/w words. BOW & TFIDF treat words as independent entities, whereas W2V creates numerical representations based upon the context in which the word occured.

GloVe → Based on Matrix Factorization of the word Co-occurance matrix

FastText → It extends W2v by representing words as character n-gram. For eg:

'where' → '<wh', 'whe', 'her', 'ere', 're>'

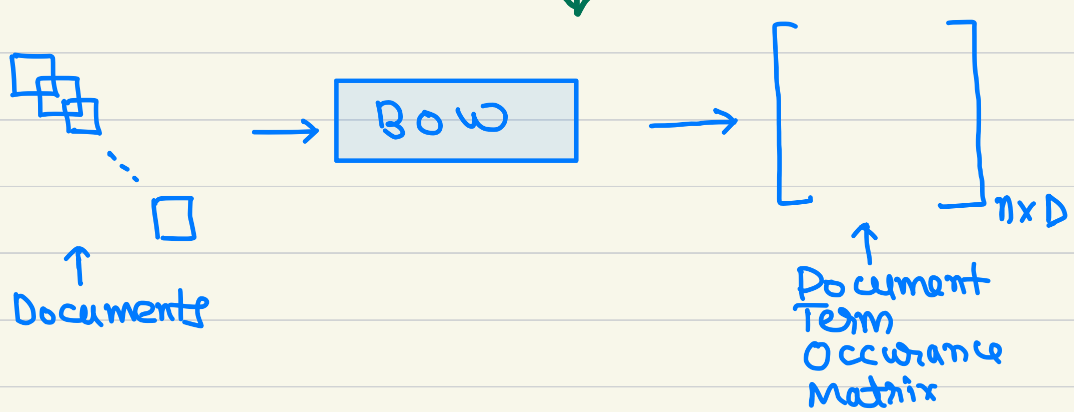
This helps capture the sub-words information & hence handles OOV words effectively.

By: Kanav Bansal
(That AI Guy)

4. ELMo → Captures contextual representation & word sequences
5. BERT → Captures contextual representation & word sequences

By: Kanav Bansal
(That AI Guy)

BoW Approach



Problems :

1. High Dimensional & Sparsity
2. Sequence is not captured
3. Can't handle OOV words
4. Semantics is not captured

Distributed Representation:

Earlier approaches were high-dim & sparse.
In distributed representation we try to compress the dimensionality which results in a compact & dense vectors.

W2v is a distributed representation.

By: Karan Bansal
(That AI Guy)

W2V

How does Word2Vec learns Embeddings?

W2V utilizes either of the following Architecture

1. CBOW
2. SkipGram
3. Skip Gram + Negative Sampling

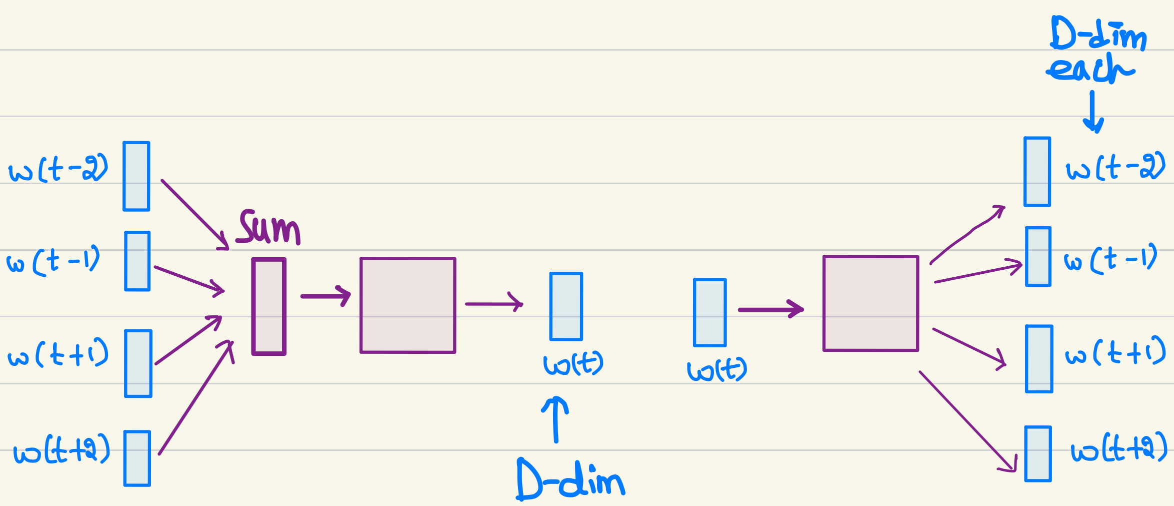
$doc_i \rightarrow \ldots\ldots\ w_5\ \ w_6\ \ w_7\ \ w_8\ \ w_9\ ----$

→ Training Data pairs for CBOW

X	y
w_5	w_7
w_6	w_7
w_8	w_7
w_9	w_7
⋮	⋮
↑ Context	↑ Target

→ Training Data pairs for SkipGram

X	y
w_7	w_5
w_7	w_6
w_7	w_8
w_7	w_9
⋮	⋮
↑ Target	↑ Context



1. CBOW

- → Not used in real time
- → Bad Representations for rare words
- → Overfits on frequent words
- → D-dim classifier is not trivial

2. SkipGram

- → Even more challenging than CBOW because with a single word now we have to predict the whole context.

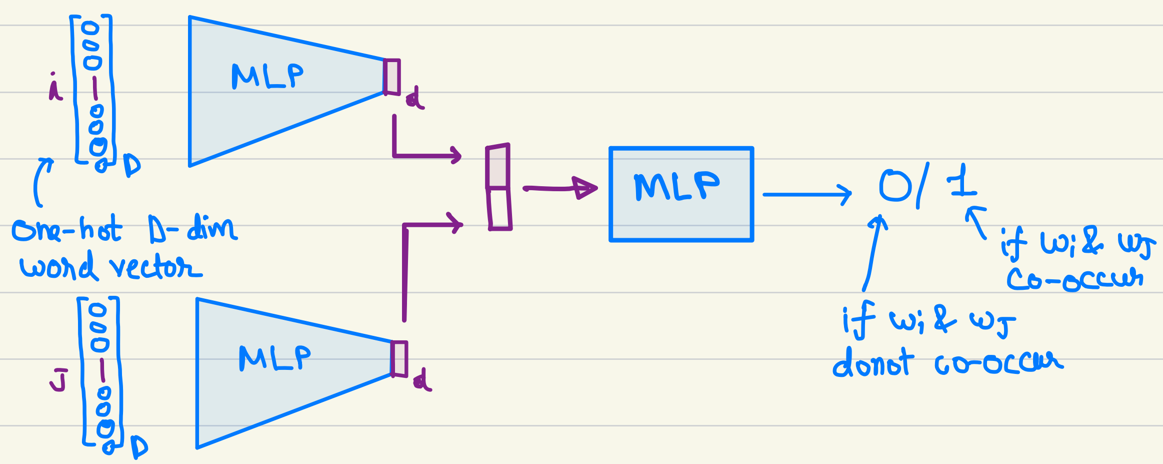
There is no negative sampling here.
We will always pass words within a context

By: Kanav Bansal
(That AI Guy)

3. Skip-Gram with Negative Sampling

Using NN to compress the dimensionality ↓

Let's learn how to take each word's D-dim one hot vector & compress it to d-dim



Skip-Gram with Negative Sampling
(Best till 2015)

A fundamental assumption here is :

If a Machine Learning architecture can start predicting if two words co-occur or not,

then the d-dim representation we get must have captured the semantic meaning of the word.

* Negative Sampling :

w_i & w_J can be the words which are not even in the context.

It's like teaching by showing both what's right & wrong.

By: Karan Bansal
(That AI Guy)

GloVe I

* Let's now have a look at Co-occurance Matrix

$doc_i \rightarrow \dots \ w_5 \ w_6 \ w_7 \ w_8 \ w_9 \ - - - -$

- Observe that w_7 cooccurs with w_5, w_6, w_8 & w_9. (consider a window of size 5)
- Now we can create a cooccurance matrix instead of document term matrix.
- This won't preserve the sequence information but some context of each word will be preserved.

$$
\begin{array}{c c}
 & \cdots \ w_J \ - - - \ w_D \\
\begin{array}{c} w_1 \\ w_2 \\ \vdots \\ w_i \\ \vdots \\ w_D \end{array} & \left[\begin{array}{ccc} & & \\ & & \\ & \bullet & \\ & & \\ & & \end{array} \right]_{D * D}
\end{array}
$$

This will be a count of how many times w_i & w_J cooccur together OR if w_i & w_J cooccur or not

Observe that the above co-occurance matrix is very large.

Distributed VS Contextual Representation

By: Karan Bansal
(That AI Guy)

Distributed Representation ⇁

Dense & Low Dimensional Representation of text

Contextual Representation ⇁

Each word can have multiple meanings based upon the context in which it is used.

Ex:

I went to a river bank for a bath. Later I went to a bank to discuss regarding my loan application.

By: Kanav Bansal
(That AI Guy)

Example of Contextual Representation & Long term Dependencies

Anaconda [redacted] 3.
4. [redacted] Amazon.
Anaconda [redacted]
[redacted] Python. 2.
1.

Can you tell the meaning of the words from above paragraph?

By: Kanav Bansal
(That AI Guy)

Example of Contextual Representation & Long term Dependencies

Anaconda can grow to be more than 30 feet & weigh over 500 pounds. 3.

They live in swamps in the rain-

4. -forests of the Amazon.

Anaconda is also a popular open source distribution of Python. 2.

1.

Can you tell the meaning of the words from above paragraph?

W2V | GloVe | FastText → Numerical Representation
captures semantics
i.e Relationships b/w words

Note: Every word will have one numerical representation.

* IDEAS:

Contextual Representation 1. One word can have multiple meanings. This meaning can be represented by looking at the near-by words OR context.

Attention 2. In order to understand the meaning we don't need to look at all the surrounding words. Just by focusing on few important words we can understand more about each word.

* Attention mechanism plays a vital role by learning context-Based Language Representations.

* Attention mechanism also help with long term dependencies.

Context Free Embeddings Summary

By: Kanav Bansal
(That AI Guy)

text corpus $\longrightarrow$ $\boxed{\text{Shallow ANN}}$ $\longrightarrow$ Word2Vec Model

How to train?
- CBOW
- SkipGram
- SkipGram with Negative Sampling

$\longrightarrow$ Helps in mapping the problem statement to classification task.

Pro:
- Dense & Low Dim Vectors
- Captures word Semantics
- FastText can capture sub-level word info & hence handling OOV words

Cons:
- Sequence info is not captured
- Contextual Representations are not captured
  (i.e. Context Free Embeddings OR Static Embeddings)

Context Based Embeddings Summary

By: Kanav Bansal
(That AI Guy)

text corpus $\longrightarrow$ $\boxed{\text{Bi-LSTM}}$ $\longrightarrow$ ELMo Model

text corpus $\longrightarrow$ $\boxed{\text{Encoder only Transformer}}$ $\longrightarrow$ BERT Model

text corpus $\longrightarrow$ $\boxed{\text{Decoder only Transformer}}$ $\longrightarrow$ GPT Model

By: Karan Bansal
(That AI Guy)

Basic Vectorization: BoW → Word occurrence; TFIDF → word importance

Distributed Representation: Word2Vec / GloVe / FastText → word's semantic

Universal Lang. Rep.: ELMo → Word's sequence + Contextual Embedding

BERT

Universal Language Representation:
- Bi-directional
- \+ Contextual embeddings
- \+ Parallel Training & Scalable
- \+ Takes care of Long-Term dependencies

↑ Using Attention Mechanism

Vectorization Techniques	BOW	TFIDF	W2V	ELMo	BERT
1. Dimensionality	↑	↑	↓	↓	↓
2. Sparsity	↑	↑	↓	↓	↓
3. Semantic	✗	✗	✓	✓	✓
4. Sequence info	✗	✗	✗	✓	✓
5. OOV	✗	✗	✗	✓	✓
6. Contextual Embeddings	✗	✗	✗	✓	✓

W2V:
Lots of Text Data
\+
Approach: CBOW / SkipGram
\+
Architecture: Shallow ANN
- No seq info
- No contextual info
- OOV can be handled by FastText

ELMo:
Lots of Text Data
\+
Approach: Language Model
\+
Architecture: Bi-LSTM
- Slow to train
- Long-Term Dependencies

BERT:
Lots of Text Data
\+
Approach: Masked LM & NSP
\+
Architecture: Encoders Only (Transformers)