Sentiment Analysis
TF-IDF

Norwegian University of Life Sciences

***Term Frequency (TF)** — How many times this word appears in a document (sentence)*
***Inverse Document Frequency (IDF)** — the natural logarithm of the total number of documents divided by (1 + the total number of documents that contain this certain word)*

Example:

```
documents = [
    "apple apple",
    "apple orange"
]
```

$$\mathrm{idf}(t,d) = \log \frac{n_d}{1+\mathrm{df}(d,t)},$$

Term Frequency (TF)

	apple	orange
0	2	0
1	1	1

IDF

$$log\frac{2}{1+2} = \text{log} = log\frac{2}{3}$$

$$log\frac{2}{1+1} = \text{log} = log\frac{2}{2}$$

https://medium.com/analytics-vidhya/understanding-tfidf-for-absolute-beginners-f2c260b8944b

Sentiment Analysis
TF-IDF

Term Frequency (TF) *— How many times this word appears in a document (sentence)*
Inverse Document Frequency (IDF) *— the natural logarithm of the total number of documents divided by (1 + the total number of documents that contain this certain word)*

Example:

```
documents = [
    "apple apple",
    "apple orange"
]
```

TF-IDF: The higher the TF-IDF score of a word, the more relevant this word is to the document.
Orange is more relevant to the document than *apple* here.

Term Frequency (TF)

	apple	orange
0	2	0
1	1	1

IDF

$$log\,\frac{2}{1+2} = \text{log} = log\,\frac{2}{3}$$

$$log\,\frac{2}{1+1} = \text{log} = log\,\frac{2}{2}$$

https://medium.com/analytics-vidhya/understanding-tfidf-for-absolute-beginners-f2c260b8944b

Sentiment Analysis
TF-IDF

Term Frequency (TF) *— How many times this word appears in a document (sentence)*
Inverse Document Frequency (IDF) *— the natural logarithm of the total number of documents divided by (1 + the total number of documents that contain this certain word)*

Example:

```
documents = [
     "apple apple",
     "apple orange"
]
```

TF: the more a word appears, the more relevant the word is to the document

Term Frequency (TF)

	apple	orange
0	2	0
1	1	1

IDF

$$log\,\frac{2}{1+2} = \text{log} = log\,\frac{2}{3}$$

$$log\,\frac{2}{1+1} = \text{log} = log\,\frac{2}{2}$$

https://medium.com/analytics-vidhya/understanding-tfidf-for-absolute-beginners-f2c260b8944b

Sentiment Analysis
TF-IDF

Term Frequency (TF) *— How many times this word appears in a document (sentence)*
Inverse Document Frequency (IDF) *— the natural logarithm of the total number of documents divided by (1 + the total number of documents that contain this certain word)*

Example:

```
documents = [
    "apple apple",
    "apple orange"
]
```

The IDF measures how rare or common a word is in the collection of documents.

- If a word appears in many documents, its IDF score will be lower, as this word is *less unique to this certain document.*
- If a word appears in fewer documents, its IDF score will be higher, as this word is *more unique to this certain document.*

Term Frequency (TF)

	apple	orange
0	2	0
1	1	1

IDF

$$log\,\frac{2}{1+2} = \text{log} = log\,\frac{2}{3}$$

$$log\,\frac{2}{1+1} = \text{log} = log\,\frac{2}{2}$$

https://medium.com/analytics-vidhya/understanding-tfidf-for-absolute-beginners-f2c260b8944b

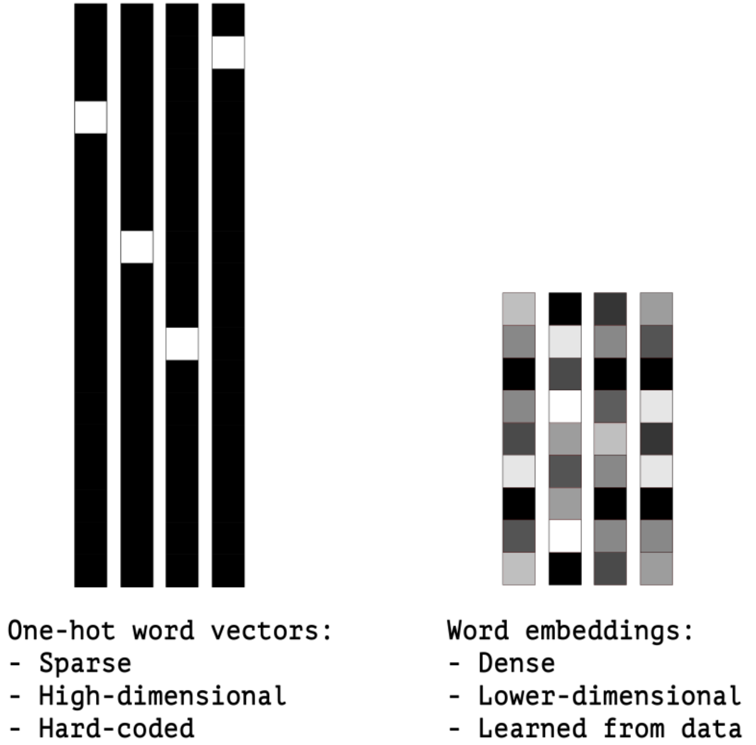
Word2Vec

- Word2Vec is a group of related models that transform words into vectors.
- Captures semantic meaning based on the context of words.

Word2Vec

- Word2Vec is a group of related models that transform words into vectors.
- Captures semantic meaning based on the context of words.

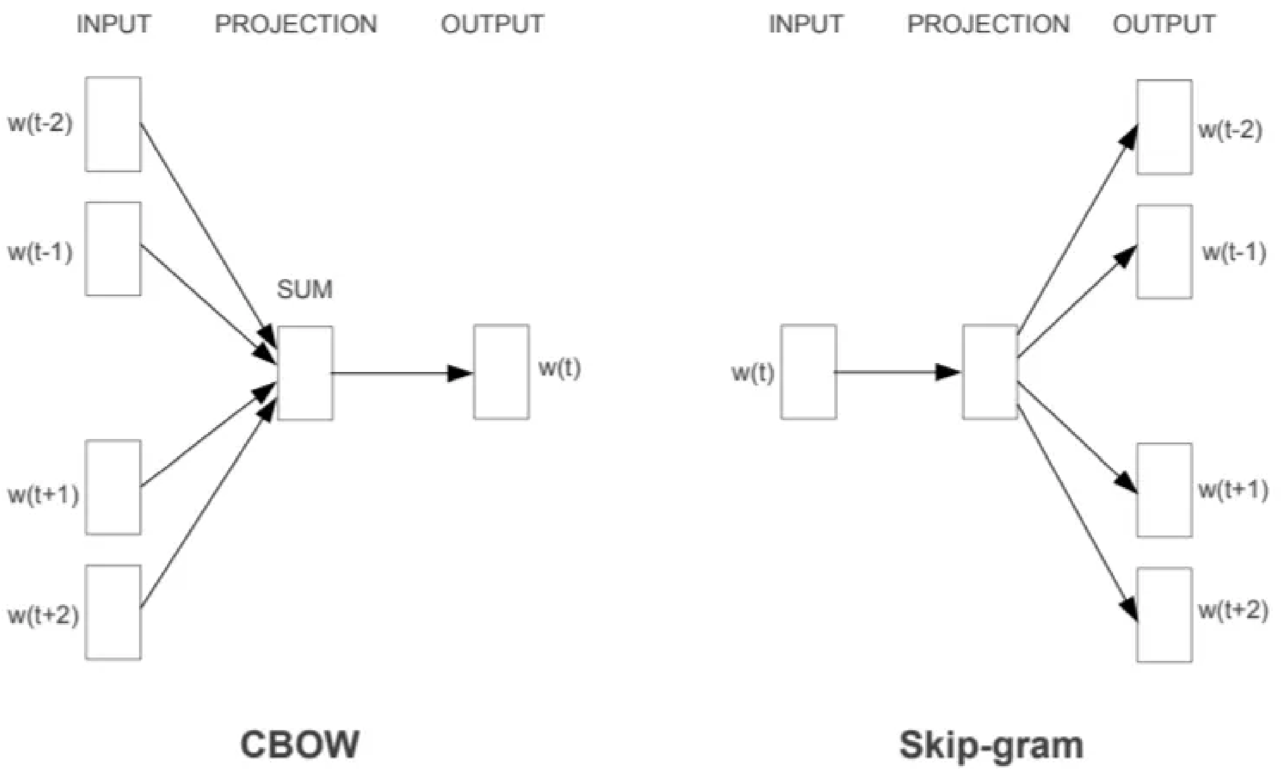
Why?
- Traditional methods represent words as discrete symbols.
- Need for capturing semantic meaning in a continuous space.



Source: https://jiajingchen.github.io/nlp/nlp-part-2/

Word2Vec Models

- Continuous Bag of Words (CBOW)
- Skip-Gram



Source: https://towardsdatascience.com/nlp-101-word2vec-skip-gram-and-cbow-93512ee24314

Continuous Bag of Words (CBOW)

- **Goal:** Given a context, predict the target word.

How?

- It takes the context (surrounding words) as input and tries to predict the word in the middle. The idea is that the semantic meaning of a word can be inferred by the words surrounding it.
- For **CBOW**, the one-hot encoded context vectors are averaged or summed, passed through the hidden layer, and then the output layer to predict the target word.

Example

Sentence: "The cat sat on the mat."

- For the word "sat", if we consider a window size of 2, the context words are ["The", "cat", "on", "the"].
- The target word is "sat".
- CBOW tries to predict "sat" given ["The", "cat", "on", "the"].

Source: https://jiajingchen.github.io/nlp/nlp-part-2/

Skip-Gram

- **Goal: G**iven a word, predict the context.

How?

- It takes a word as input and tries to predict the context, meaning the surrounding words. It's believed to work better with a small amount of data and with rare words.
- For **Skip-Gram**, the one-hot encoded word vector is passed through the hidden layer and then the output layer to predict context words.

Example

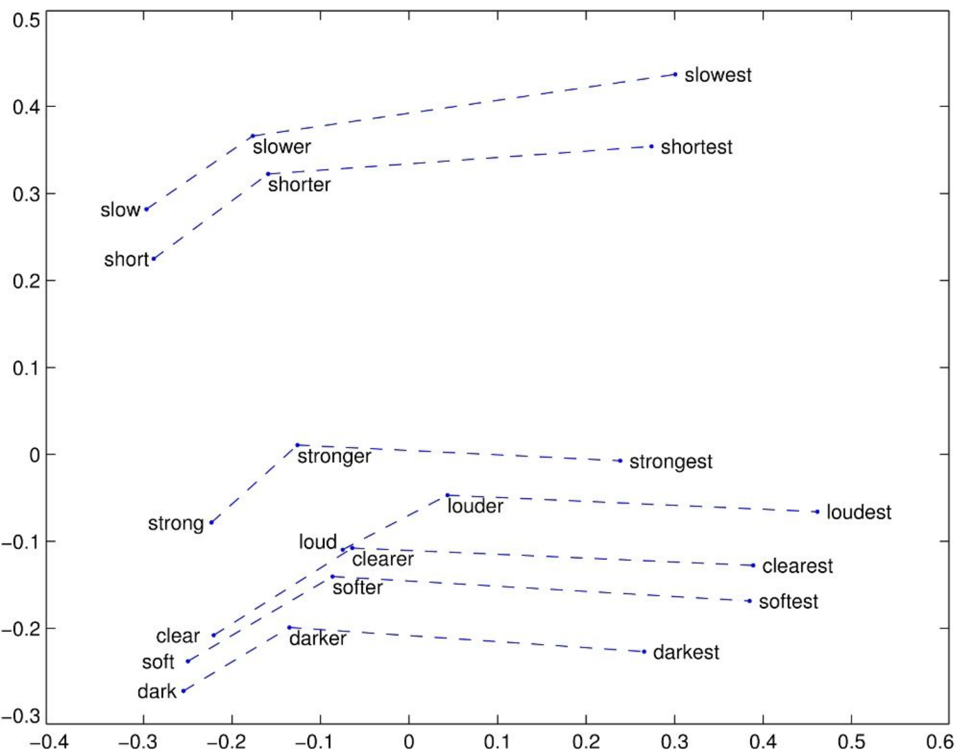
Sentence: "The cat sat on the mat."

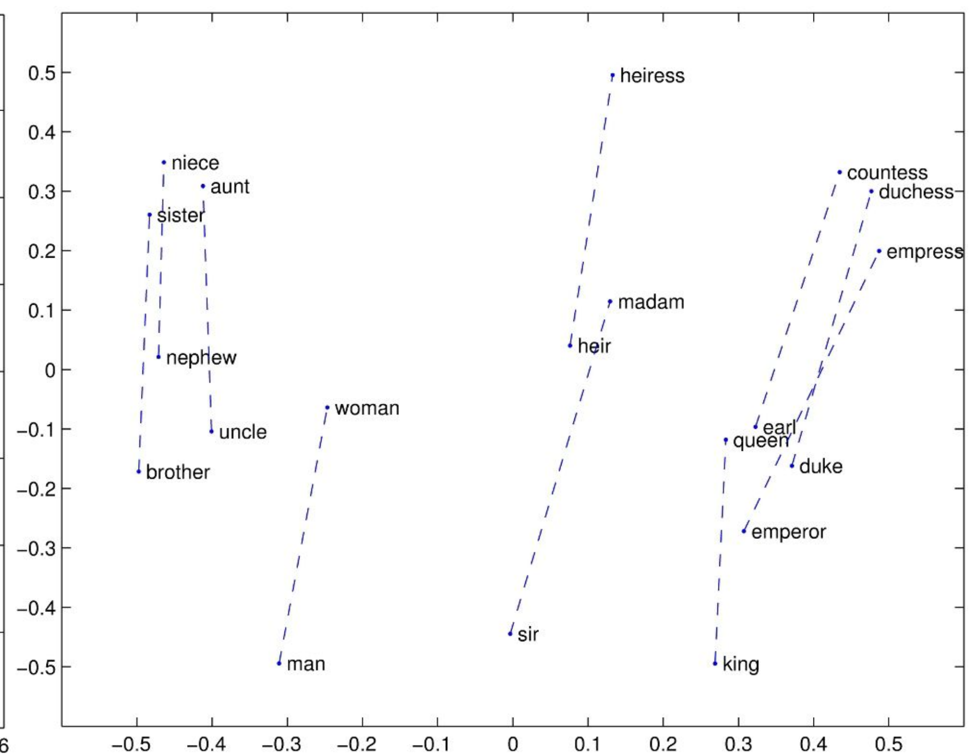
- For the word "sat", with a window size of 2, the target words are ["The", "cat", "on", "the"].
- Skip-Gram tries to predict each of these context words given the input word "sat".

Source: https://jiajingchen.github.io/nlp/nlp-part-2/

Differences

- **Data Efficiency:** Skip-Gram usually works well with less data and captures more detailed word relationships.
- **Performance:** CBOW is faster since it averages over the context words, but Skip-Gram generally has superior quality, especially with infrequent words.





Source: https://www.ruder.io/secret-word2vec/