N-grams in NLP

Abhishek Jain · Follow

3 min read · Feb 5, 2024

N-grams, a fundamental concept in NLP, play a pivotal role in capturing patterns and relationships within a sequence of words. In this blog post, we'll delve into the world of N-grams, exploring their significance, applications, and how they contribute to enhancing language processing tasks.

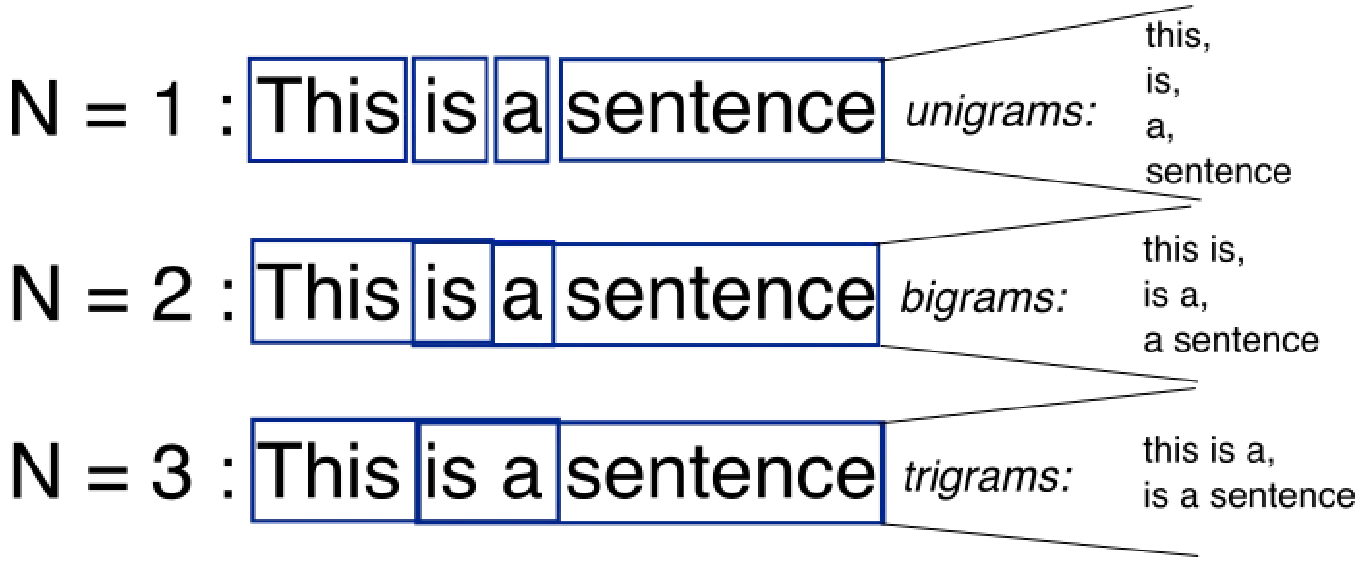
Understanding N-grams:

Definition:

N-grams are contiguous sequences of 'n' items, typically words in the context of NLP. These items can be characters, words, or even syllables, depending on the granularity desired. The value of 'n' determines the order of the N-gram.

Examples:

- **Unigrams (1-grams):** Single words, e.g., "cat," "dog."
- **Bigrams (2-grams):** Pairs of consecutive words, e.g., "natural language," "deep learning."
- **Trigrams (3-grams):** Triplets of consecutive words, e.g., "machine learning model," "data science approach."
- 4-grams, 5-grams, etc.: Sequences of four, five, or more consecutive words.



Significance of N-grams in NLP:

1. Capturing Context and Semantics:

- N-grams help capture the contextual information and semantics within a sequence of words, providing a more nuanced understanding of language.

2. Improving Language Models:

- In language modeling tasks, N-grams contribute to building more accurate and context-aware models, enhancing the performance of applications such as machine translation and speech recognition.

3. Enhancing Text Prediction:

- N-grams are essential for predictive text applications, aiding in the prediction of the next word or sequence of words based on the context provided by the preceding N-gram.

4. Information Retrieval:

- In information retrieval tasks, N-grams assist in matching and ranking documents based on the relevance of N-gram patterns.

5. Feature Extraction:

- N-grams serve as powerful features in text classification and sentiment analysis, capturing meaningful patterns that contribute to the characterization of different classes or sentiments.

Applications of N-grams in NLP:

1. Speech Recognition:

- N-grams play a crucial role in modeling and recognizing spoken language patterns, improving the accuracy of speech recognition systems.

2. Machine Translation:

- In machine translation, N-grams contribute to understanding and translating phrases within a broader context, enhancing the overall translation quality.

3. Predictive Text Input:

- Predictive text input on keyboards and mobile devices relies on N-grams to suggest the next word based on the context of the input sequence.

4. Named Entity Recognition (NER):

- N-grams aid in identifying and extracting named entities from text, such as names of people, organizations, and locations.

5. Search Engine Algorithms:

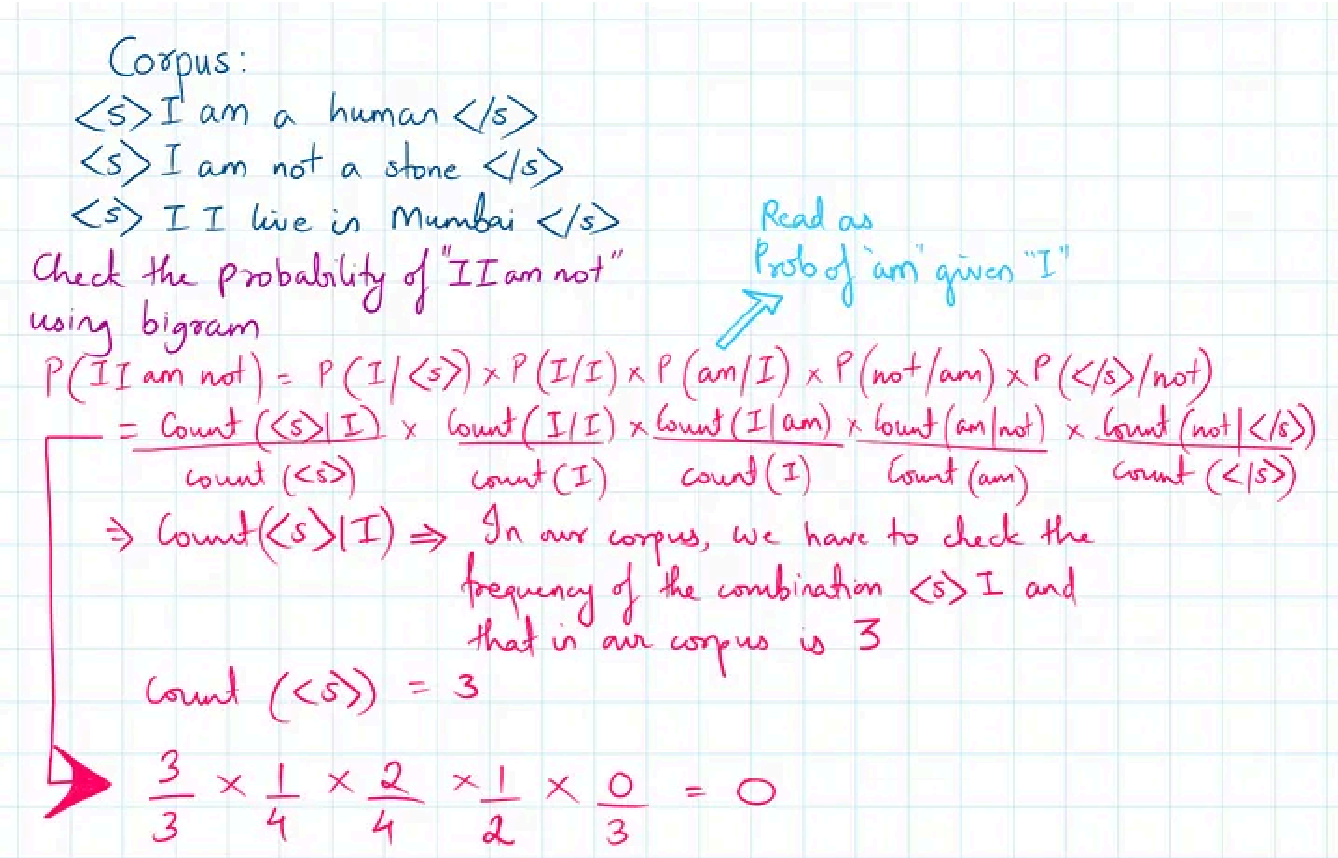
- Search engines use N-grams to index and retrieve relevant documents based on user queries, improving the accuracy of search results.

CODE

```
import nltk
nltk.download('punkt')

from nltk import ngrams
from nltk.tokenize import word_tokenize

# Example sentence
sentence = "N-grams enhance language processing tasks."

# Tokenize the sentence
tokens = word_tokenize(sentence)

# Generate bigrams
bigrams = list(ngrams(tokens, 2))

# Generate trigrams
trigrams = list(ngrams(tokens, 3))

# Print the results
print("Bigrams:", bigrams)
print("Trigrams:", trigrams)

'''
Output:
```

```
Bigrams: [('N-grams', 'enhance'), ('enhance', 'language'), ('language', 'proces
Trigrams: [('N-grams', 'enhance', 'language'), ('enhance', 'language', 'process
'''
```

Using N Gram to predict the probability of a sentence



Using N grams to predict the next word in the sentence

Consider the following training data

<s> I am Jack </s>
<s> Jack I am </s>
<s> Jack I like </s>
<s> Jack I do like </s>
<s> do I like Jack </s>

Assume that we use a bigram language model based on the above data

What is the most probable next word predicted by model

1) <s> Jack ______
2) <s> Jack I do ______
3) <s> Jack I am Jack ______
4) <s> do I like ______

$$P(I \mid \langle s \rangle) = \frac{\text{Count}(\langle s \rangle \mid I)}{\text{Count}(\langle s \rangle)} = \frac{1}{5}$$

$$P(\text{am} \mid I) = \frac{\text{Count}(I \mid \text{am})}{\text{Count}(I)} = \frac{2}{5}$$

$$P(\text{Jack} \mid \text{am}) = \frac{\text{Count}(\text{am} \mid \text{Jack})}{\text{Count}(\text{am})} = \frac{1}{2}$$

$$P(\langle /s \rangle \mid \text{Jack}) = \frac{\text{Count}(\text{Jack} \mid \langle /s \rangle)}{\text{Count}(\text{Jack})} = \frac{2}{5}$$

$$P(\text{Jack} \mid \langle s \rangle) = \frac{\text{Count}(\langle s \rangle \mid \text{Jack})}{\text{Count}(\langle s \rangle)} = \frac{3}{5}$$

$$P(I \mid \text{Jack}) = \frac{\text{Count}(\text{Jack} \mid I)}{\text{Count}(\text{Jack})} = \frac{3}{5}$$

$$P(\langle /s \rangle \mid \text{am}) = \frac{\text{Count}(\text{am} \mid \langle /s \rangle)}{\text{Count}(\text{am})} = \frac{1}{2}$$

$$P(\text{like} \mid I) = \frac{\text{Count}(I \mid \text{like})}{\text{Count}(I)} = \frac{2}{5}$$

$$P(\langle /s \rangle \mid \text{like}) = \frac{\text{Count}(\text{like} \mid \langle /s \rangle)}{\text{Count}(\text{like})} = \frac{2}{3}$$

$$P(\text{do} \mid I) = \frac{\text{Count}(I \mid \text{do})}{\text{Count}(I)} = \frac{1}{5}$$

$$P(\text{like} \mid \text{do}) = \frac{\text{Count}(\text{do} \mid \text{like})}{\text{Count}(\text{do})} = \frac{1}{2}$$

$$P(\text{do} \mid \langle s \rangle) = \frac{\text{Count}(\langle s \rangle \mid \text{do})}{\text{Count}(\langle s \rangle)} = \frac{1}{5}$$

$$P(I \mid \text{do}) = \frac{\text{Count}(\text{do} \mid I)}{\text{Count}(\text{do})} = \frac{1}{2}$$

$$P(\text{Jack} \mid \text{like}) = \frac{\text{Count}(\text{like} \mid \text{Jack})}{\text{Count}(\text{like})} = \frac{1}{3}$$

1) Jack ____

$\Rightarrow P(\text{something}|\text{Jack})$ = In our calculated probabilites we got 2 probabilities

1) $P(</s>|\text{Jack}) = \frac{2}{5}$

2) $P(I|\text{Jack}) = \frac{3}{5}$

$\Big\}$ Since $\frac{3}{5} > \frac{2}{5}$, I is the next word

2) Jack I do ____

$P(\text{something}|\text{do}) \rightarrow P(I|\text{do}) = 1/2$

$\hookrightarrow P(\text{like}|\text{do}) = 1/2$

$\Big\}$ The answer is both I and like

Ngrams Unigram Bigrams Trigram Naturallanguageprocessing

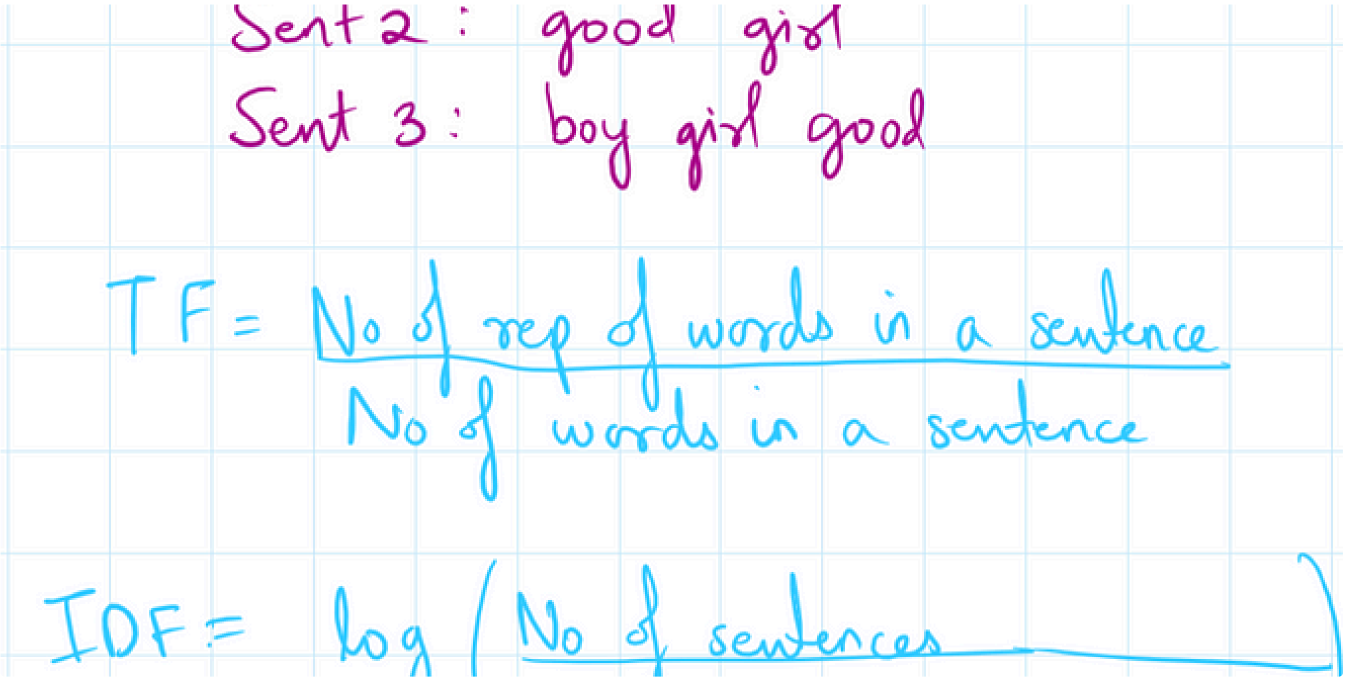
Follow

Written by Abhishek Jain

214 Followers

Data Science Enthusiast || AI-ML Engineer Linked in : https://www.linkedin.com/in/abhishek-jain-pict-aiml-enthusiast/

More from Abhishek Jain

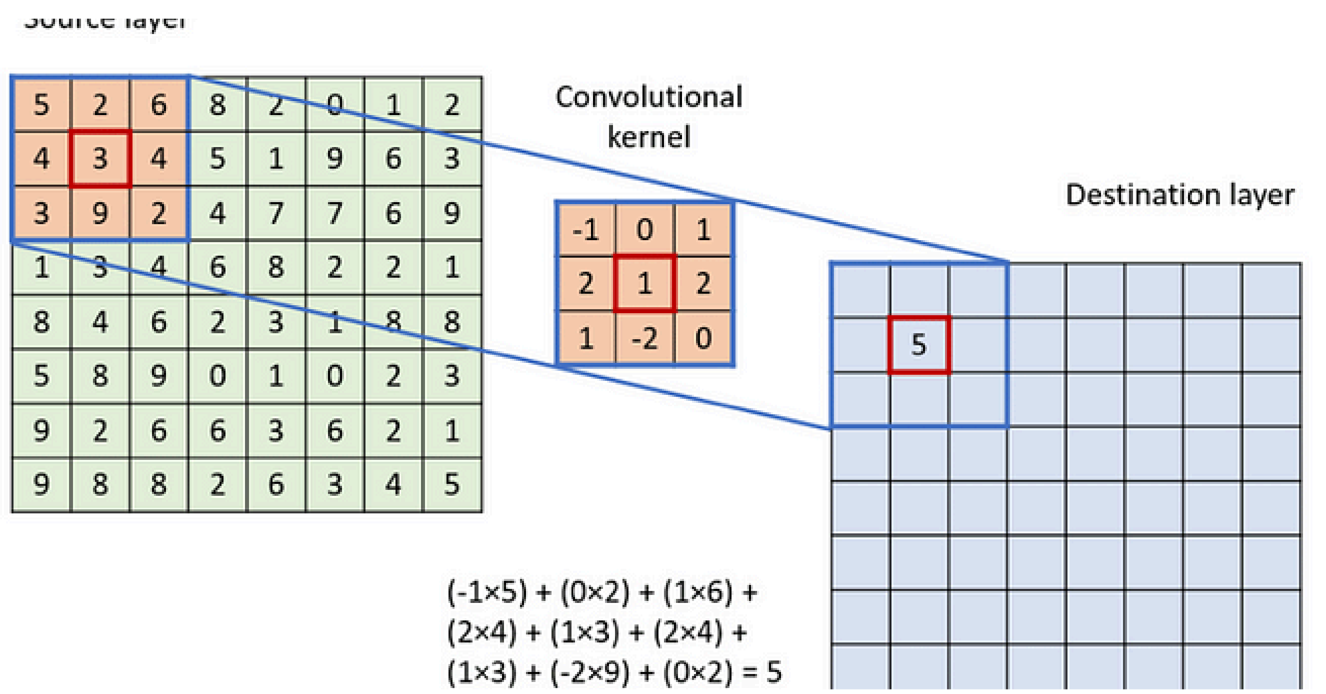


Abhishek Jain

TF-IDF in NLP (Term Frequency Inverse Document Frequency)

In the realm of Natural Language Processing (NLP), TF-IDF (Term Frequency-Inverse Document Frequency) is a powerful technique used to…

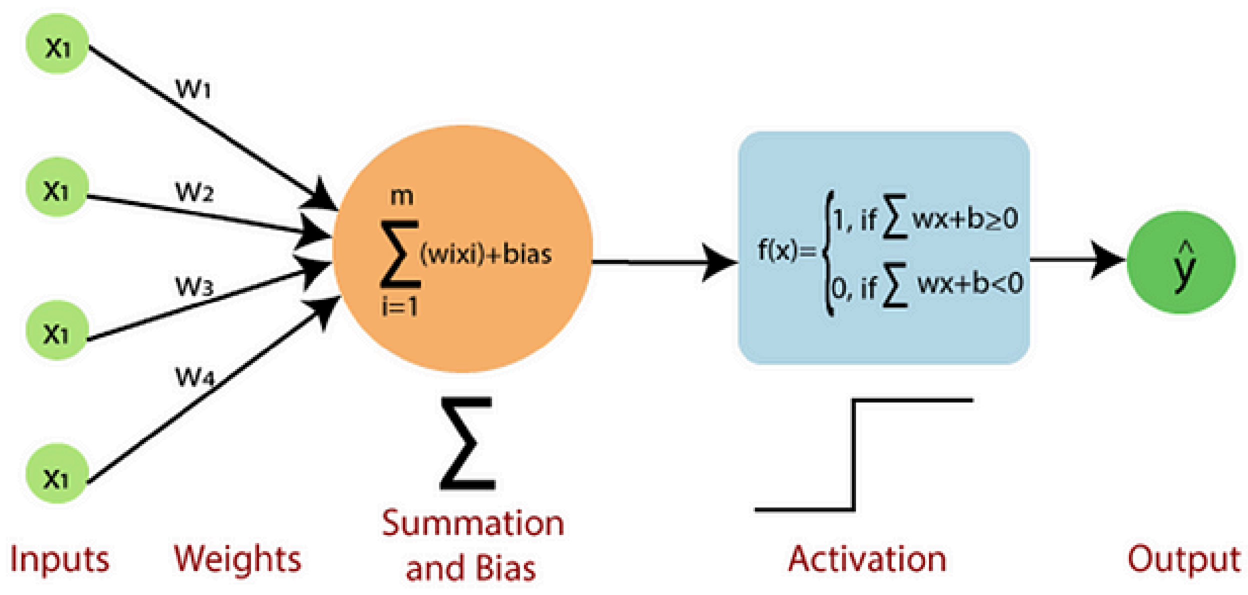
Feb 4 18



Abhishek Jain

All about convolutions, kernels, features in CNN

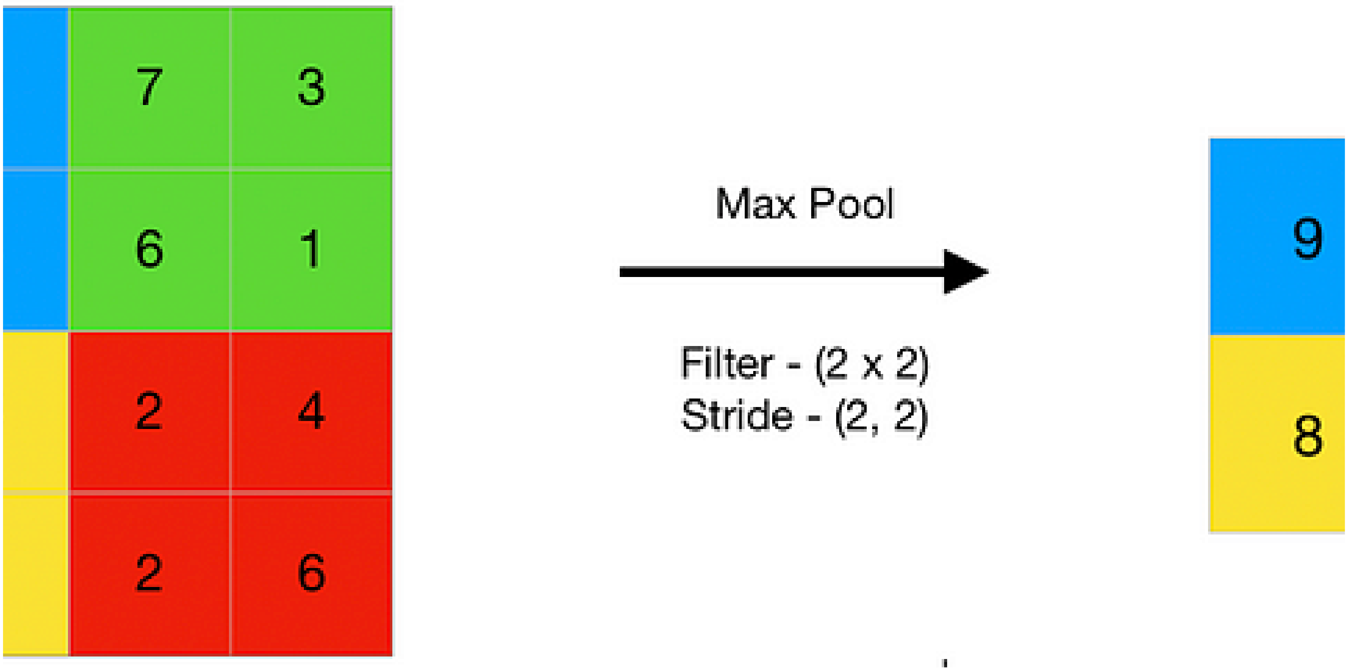
Convolution Kernels:



Abhishek Jain

Perceptron vs neuron, Single layer Perceptron and Multi Layer Perceptron

In deep learning, the terms “perceptron” and “neuron” are related but have distinct meanings, and they are not exactly the same. While both…



Abhishek Jain

Pooling and their types in CNN

Pooling is a fundamental operation in Convolutional Neural Networks (CNNs) that plays a crucial role in downsampling feature maps while...

Feb 12 14 1

See all from Abhishek Jain

Recommended from Medium

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i$$

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2$$

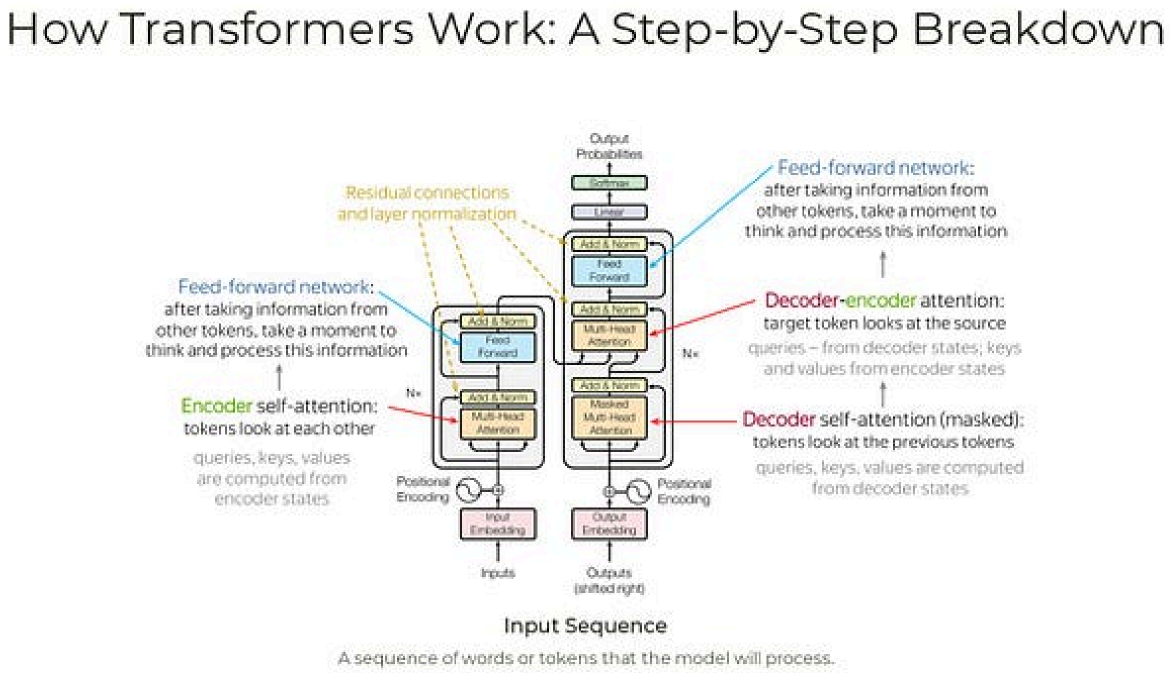
Here, x_i are the activations in the mini-batch, μ_B is the mean, σ_B^2 is the variance, and m is the mini-batch size.

J Jo Wang

Deep Learning Part 5 -How to prevent overfitting

Techniques used to prevent overfitting in deep learning models:

Jun 29 1 1



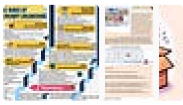
Asad Ali

Understanding the Transformer Architecture in LLM

How Transformers Work: A Step-by-Step Breakdown

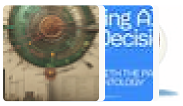
Jun 15 102

Lists



Natural Language Processing

1815 stories · 1428 saves



data science and AI

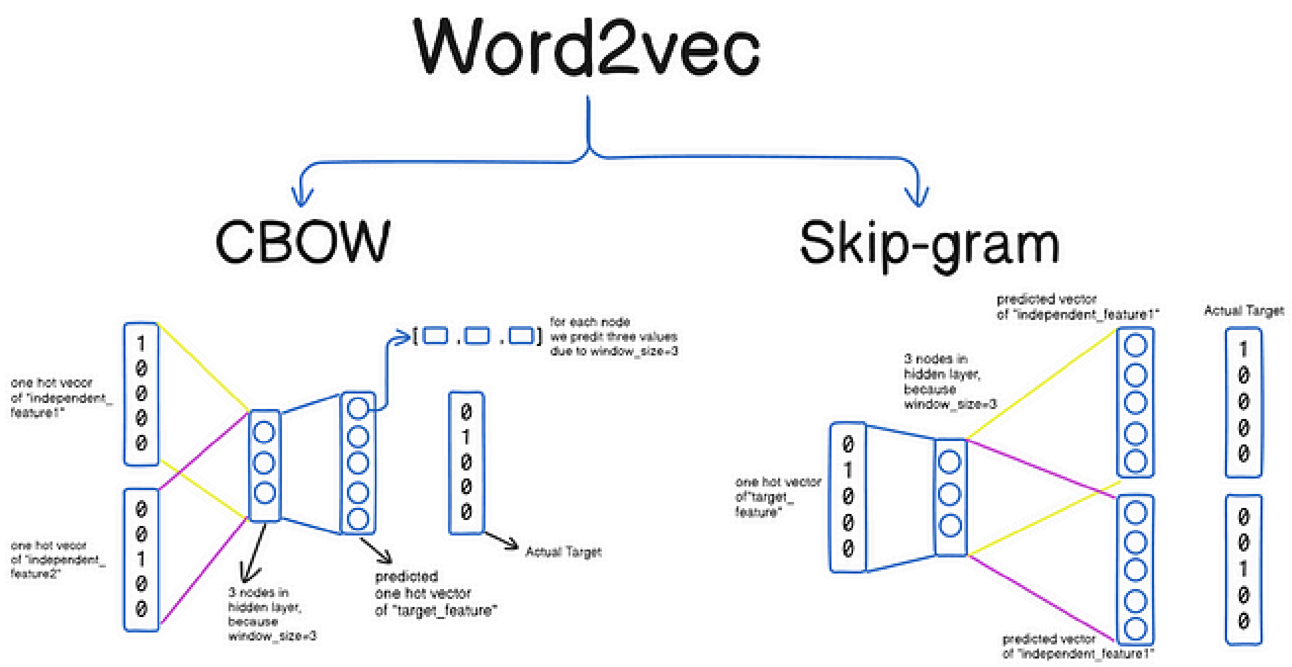
40 stories · 286 saves

Abdur Rahman in Stackademic

Python is No More The King of Data Science

5 Reasons Why Python is Losing Its Crown

Oct 23 · 6.8K · 29

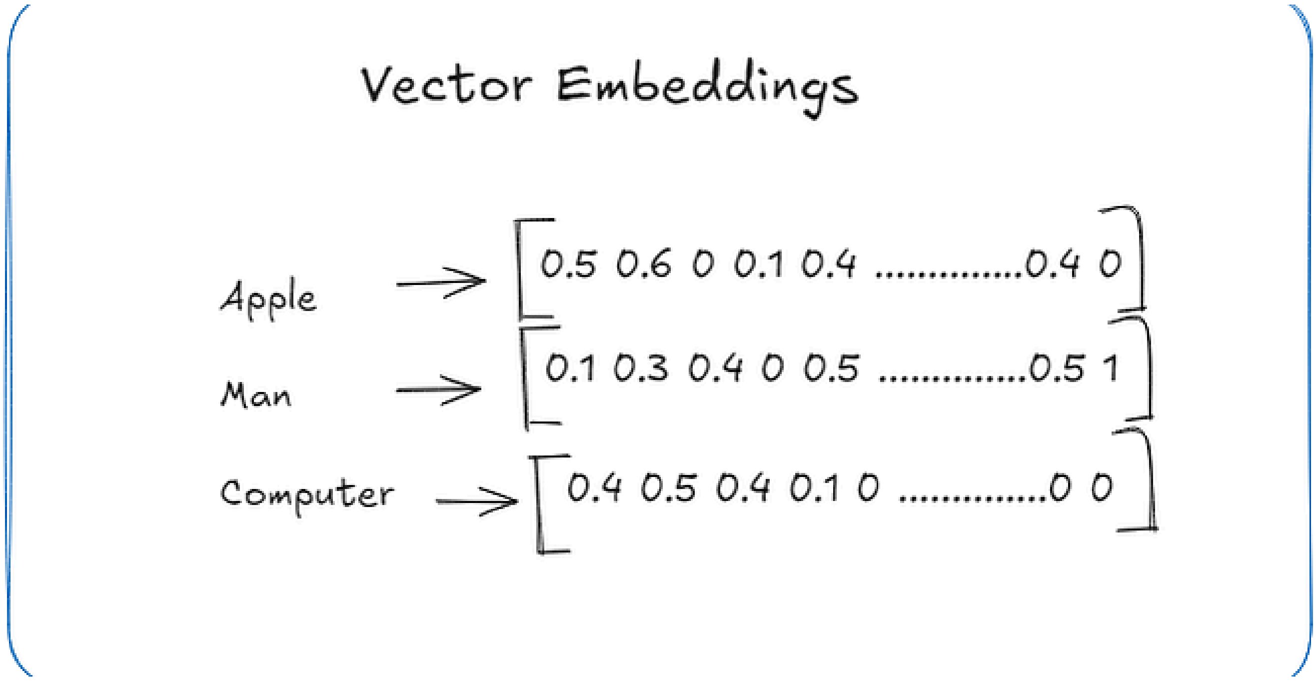


Fraidoon Omarzai

Word2Vec (CBOW, Skip-gram) In Depth

Word2Vec is an important model for natural language processing (NLP) developed by researchers at Google.

Aug 1 2



Mdabdullahalhasib in Towards AI

A Complete Guide to Embedding For NLP & Generative AI/LLM

Understand the concept of vector embedding, why it is needed, and implementation with LangChain.

Oct 19 110

Jo Wang

Deep Learning Part 2—Neural Network and the critical Activation Functions

Neural Network Structure

Jun 29 1

See more recommendations