

Indonesia AI

Frequency-Based Text Vectorization

Proprietary document of Indonesia AI 2023

OBJECTIVE & OUTLINE

Proprietary document of Indonesia AI 2023



Frequency-Based Text Vectorization

Objektif: Memahami konsep dari Frequency-Based Text Vectorization dalam NLP

Outline:

1. Konsep Frequency-Based Text Vectorization
2. Count Vector
3. TF-IDF
4. Co-Occurance Matrix
5. N-Gram

Apa itu Frequency-Based Text Vectorization?

TEXT VECTORIZATION

Proprietary document of Indonesia AI 2023



Proses di mana sekumpulan **teks** akan **diubah** menjadi **representasi numerik** sehingga dapat dimengerti oleh komputer,

TEXT VECTORIZATION

Proprietary document of Indonesia AI 2023



Kenapa Text Vectorization Penting?

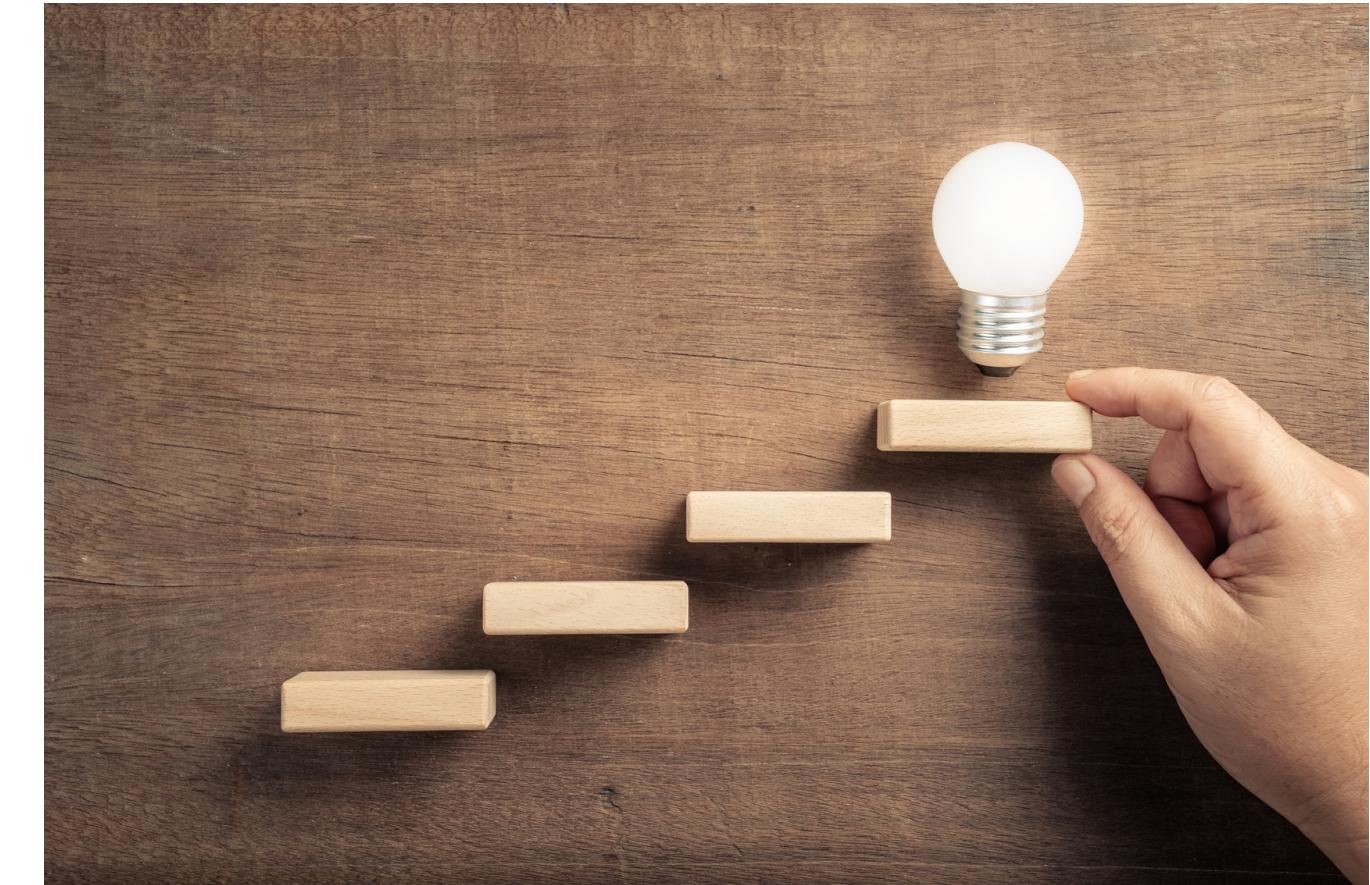
- Representasi Numerik
- Ekstraksi Fitur
- Reduksi Dimensi
- Kompatibilitas Model

TEXT VECTORIZATION

Proprietary document of Indonesia AI 2023

Text Vectorization Pada Preprocessing

- Tokenisasi kalimat
- Menghilangkan *stopwords*
- Normalisasi kata
- Text Vectorization



TEXT VECTORIZATION

Proprietary document of Indonesia AI 2023



Macam Text Vectorization

- Frequency-Based Text Vectorization
- Prediction-Based Text Vectorization

FREQUENCY-BASED TEXT VECTORIZATION

Proprietary document of Indonesia AI 2023



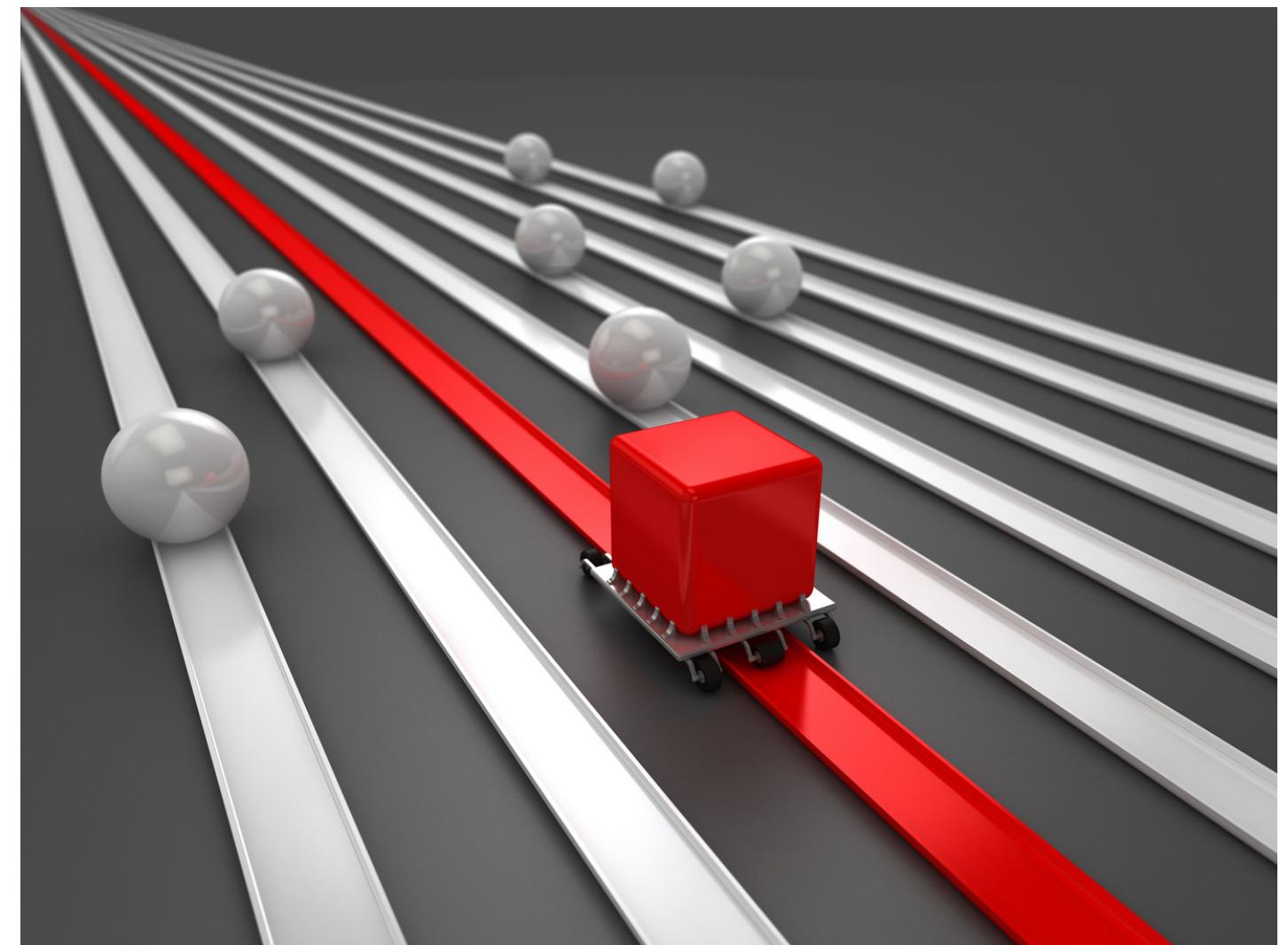
Metode yang digunakan untuk mengonversikan **teks** ke dalam bentuk **vektor numerik** berdasarkan **frekuensi kemunculan** kata-kata dalam teks tersebut.

FREQUENCY-BASED TEXT VECTORIZATION

Proprietary document of Indonesia AI 2023

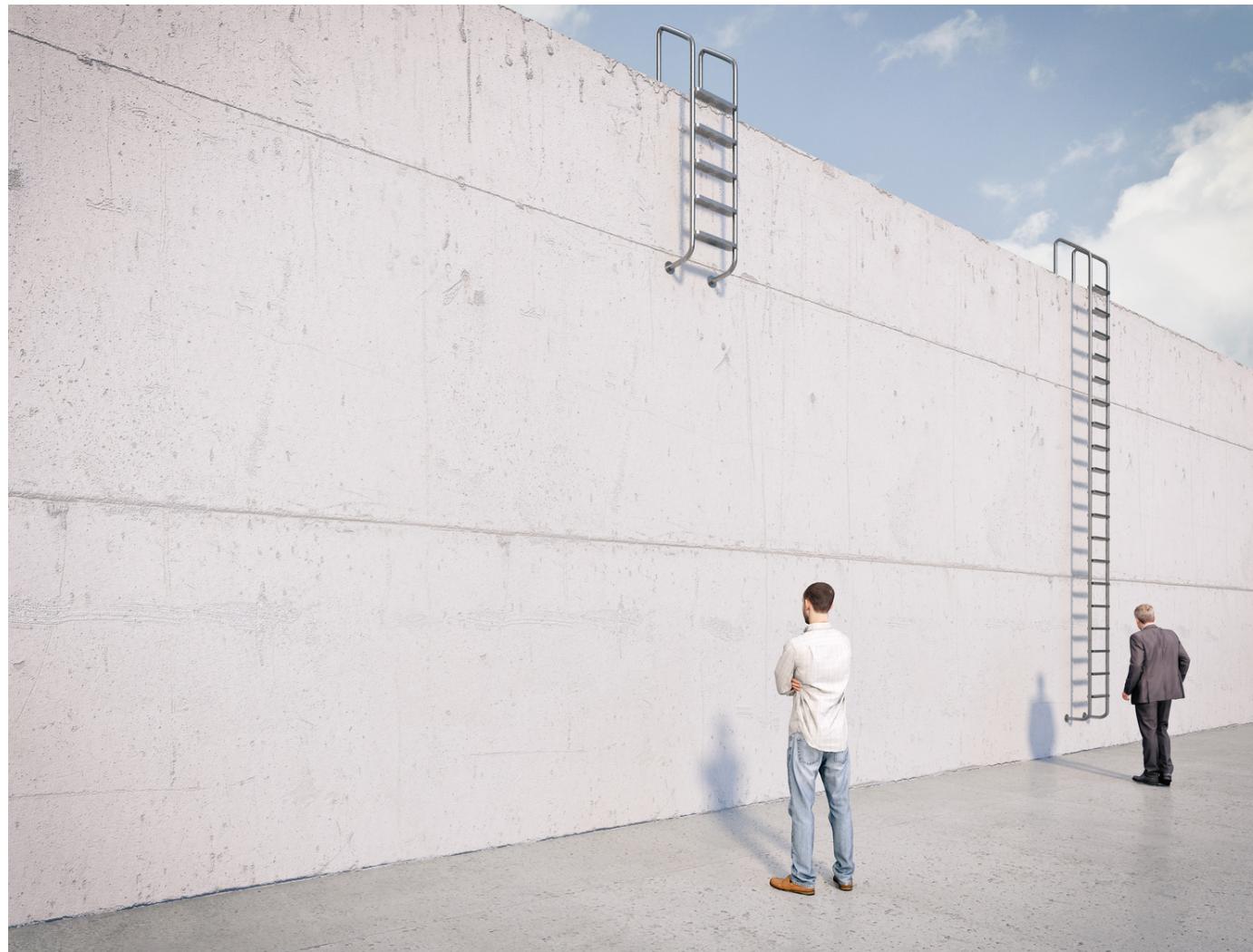
Kelebihan Frequency-Based Text Vectorization

- Sederhana dan mudah dipahami
- Menjaga informasi frekuensi
- Kompabilitas dengan model klasik



FREQUENCY-BASED TEXT VECTORIZATION

Proprietary document of Indonesia AI 2023



Kekurangan Frequency-Based Text Vectorization

- Tidak memperhitungkan konteks
- Sensitif terhadap skala
- Tidak dapat menangani urutan kata

FREQUENCY-BASED TEXT VECTORIZATION

Proprietary document of Indonesia AI 2023

Macam Frequency-Based Text Vectorization



- Count Vector
- TF-IDF
- Co-Occurrence Matrix
- N-Gram

— Any question guys ~

— Count Vector

COUNT VECTOR

Proprietary document of Indonesia AI 2023

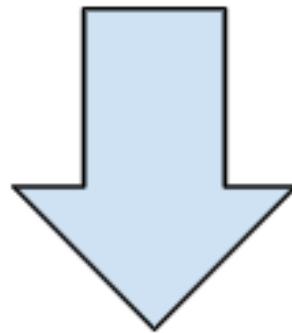
Count Vector menghitung **frekuensi kemunculan** setiap **kata dalam teks** dan membuat vektor dengan menggunakan skema **frekuensi kata**

COUNT VECTOR

Proprietary document of Indonesia AI 2023

```
dokumen = ["satu topi, dua topi, topi baru, semua tentang topi"]
```

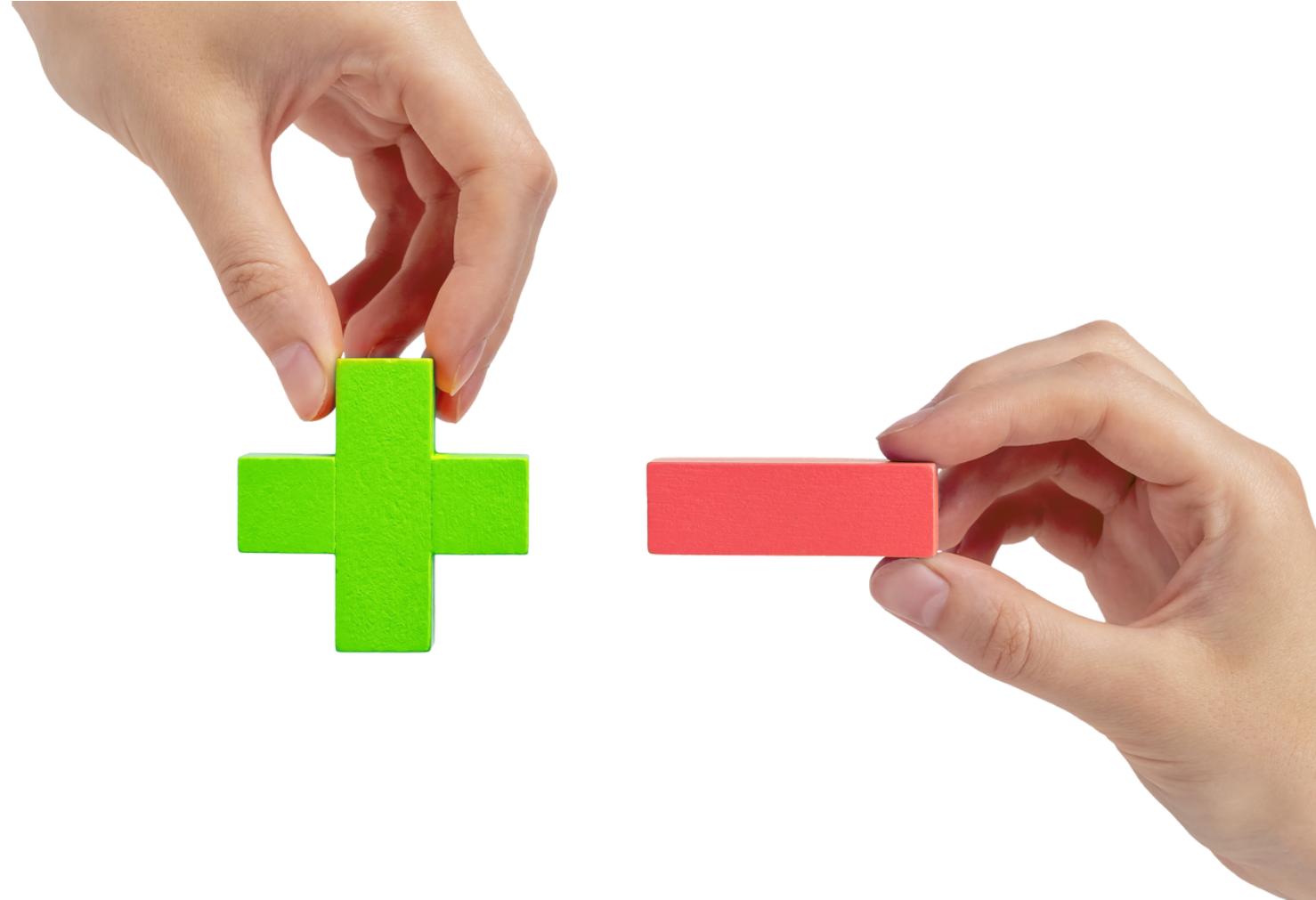
	satu	topi	dua	baru	semua	tentang
dokumen	1	4	1	1	1	1



index	0	1	2	3	4	5
dokumen	1	4	1	1	1	1

COUNT VECTOR

Proprietary document of Indonesia AI 2023



Kelebihan

- Sederhana, mudah diimplementasikan
- Mempertahankan informasi kemunculan data

Kekurangan

- Tidak mempertimbangkan bobot kata
- Tidak mengatasi isu kata umum yang mungkin kurang relevan

— Any question guys ~

— TF-IDF

TF-IDF (Term Frequency-Inverse Document Frequency) memberikan **bobot** pada kata berdasarkan **frekuensi kemunculan kata** dalam suatu teks dan seberapa umum kata tersebut dalam **keseluruhan teks**

Dokumen 1: "Saya suka makan nasi"

Dokumen 2: "Saya suka makan kentang"

Hitung **Term Frequency (TF)** untuk setiap kata dalam setiap dokumen. TF adalah rasio jumlah kemunculan kata tersebut dengan total kata dalam dokumen.

Dokumen 1:

- $\text{TF}(\text{"saya"}) = 1/4 = 0.25$
- $\text{TF}(\text{"suka"}) = 1/4 = 0.25$
- $\text{TF}(\text{"makan"}) = 1/4 = 0.25$
- $\text{TF}(\text{"nasi"}) = 1/4 = 0.25$

Dokumen 2:

- $\text{TF}(\text{"saya"}) = 1/4 = 0.25$
- $\text{TF}(\text{"suka"}) = 1/4 = 0.25$
- $\text{TF}(\text{"makan"}) = 1/4 = 0.25$
- $\text{TF}(\text{"kentang"}) = 1/4 = 0.25$

Hitung **Inverse Document Frequency (IDF)** untuk setiap kata. IDF adalah logaritma dari jumlah total dokumen dibagi dengan jumlah dokumen yang mengandung kata tersebut.

- $\text{IDF}(\text{"saya"}) = \log(2/2) = 0$
- $\text{IDF}(\text{"suka"}) = \log(2/2) = 0$
- $\text{IDF}(\text{"makan"}) = \log(2/2) = 0$
- $\text{IDF}(\text{"nasi"}) = \log(2/1) = 0.3$
- $\text{IDF}(\text{"kentang"}) = \log(2/1) = 0.3$

Kalikan TF dengan IDF untuk mendapatkan nilai TF-IDF untuk setiap kata dalam setiap dokumen.

Dokumen 1:

- $TF("saya") = 0.25 * 0 = 0$
- $TF("suka") = 0.25 * 0 = 0$
- $TF("makan") = 0.25 * 0 = 0$
- $TF("nasi") = 0.25 * 0.3 = 0.075$

Dokumen 2:

- $TF("saya") = 0.25 * 0 = 0$
- $TF("suka") = 0.25 * 0 = 0$
- $TF("makan") = 0.25 * 0 = 0$
- $TF("kentang") = 0.25 * 0.3 = 0.075$

Kelebihan

- Memperhitungkan pentingnya kata berdasarkan frekuensi kemunculan
- Mengurangi bobot kata-kata umum yang tidak informatif

Kekurangan

- Tidak memperhitungkan hubungan antara kata-kata yang muncul bersama
- Mungkin tidak cocok untuk teks dengan konteks yang kompleks



— Any question guys ~

— Co-Occurance Matrix

CO-OCCURANCE MATRIX

Proprietary document of Indonesia AI 2023

Co-Occurrence Matrix menghitung **seberapa sering kata-kata muncul bersama** dalam teks dan membuat matriks yang **merepresentasikan hubungan** antara kata-kata.

CO-OCCURANCE MATRIX

Proprietary document of Indonesia AI 2023

Dokumen 1: "Saya suka makan nasi"

Dokumen 2: "Saya suka makan kentang"

CO-OCCURANCE MATRIX

Proprietary document of Indonesia AI 2023

	saya	suka	makan	nasi	kentang
saya					
suka					
makan					
nasi					
kentang					

CO-OCCURANCE MATRIX

Proprietary document of Indonesia AI 2023

	saya	suka	makan	nasi	kentang
saya	0	2	2	1	1
suka	2	0	2	1	1
makan	2	2	0	1	1
nasi	1	1	1	0	0
kentang	1	1	1	0	0

CO-OCCURANCE MATRIX

Proprietary document of Indonesia AI 2023



Kelebihan

- Memperhitungkan hubungan antara kata-kata
- Cocok untuk mendeteksi hubungan semantik

Kekurangan

- Matriks yang dihasilkan dapat menjadi besar
- Memerlukan proses lebih lanjut seperti reduksi dimensi

— Any question guys ~

N-GRAM



N-Gram **membagi teks** menjadi **urutan N kata**, dan membuat vektor yang **merepresentasikan kemunculan urutan kata tersebut**



Dokumen: "saya suka menikmati film"



Unigram: Unigram adalah N-Gram dengan N=1, yang membagi teks menjadi kata-kata tunggal.

Dokumen: ["saya", "suka", "menikmati", "film"]



Bigram: Bigram adalah N-Gram dengan N=2, yang membagi teks menjadi pasangan kata berurutan.

Dokumen: ["saya suka", "suka menikmati", "menikmati film"]



Trigram: Trigram adalah N-Gram dengan N=3, yang membagi teks menjadi tiga kata berurutan.

Dokumen: ["saya suka menikmati", "suka menikmati film"]

Kelebihan

- Memperhitungkan urutan kata dalam teks
- Cocok untuk tugas seperti prediksi kata berikutnya atau deteksi frasa

Kekurangan

- Rentan terhadap dimensi yang tinggi
- Tidak mempertimbangkan informasi kata-kata individual



— Any question guys ~

Terima Kasih!