

Welcome Everyone



Nama

Informasi tambahan

 email{at}orbitfutureacademy.sch.id

 Linkedin / web

Our Rules – NLP Squad

Kehadiran:

- Toleransi keterlambatan 15 menit.
- Izin kehadiran memberikan konfirmasi dengan alasan yang penting kepada homeroom coach.

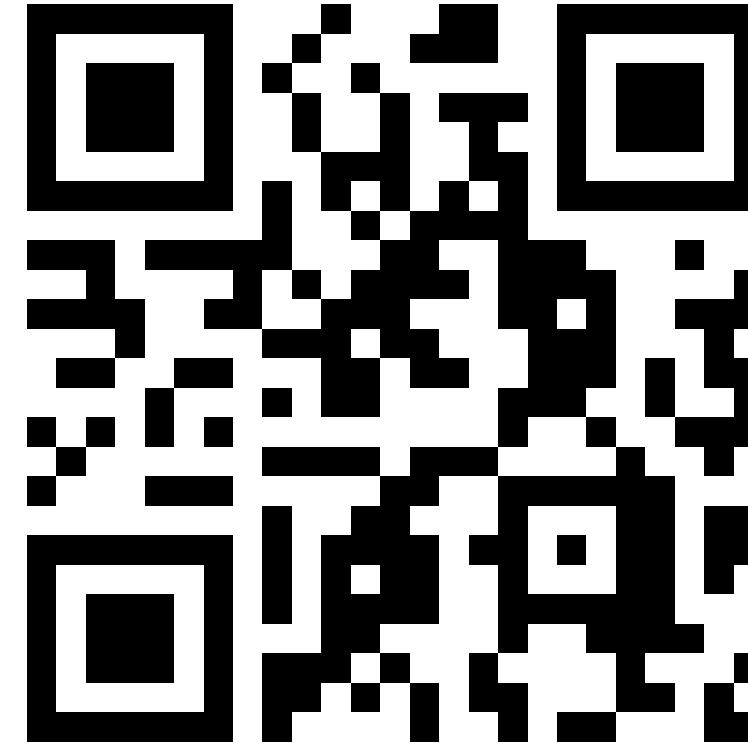
General Rule:

- *Positive attitude.* Hargai semua orang di kelas ini.
- Aktif dalam pembelajaran.
- Jika terjadi hal di luar kendali seperti mati listrik dan lain sebagainya, lakukan konfirmasi via chat.
- Rename nama zoom: NamaKelas_NamaLengkap.

Agenda Khusus:

- Mengisi logbook pada web Kampus Merdeka.
- Mengisi presensi dan feedback form.

Pre-Test NLP



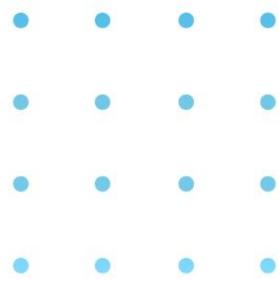
<https://s.id/pretest-nlp>

Learning Objective – NLP Squad

1. Introduction to NLP
2. Text Representation
3. Text Classification
4. Text Summarization
5. Word Embedding Techniques
6. Deep Learning in NLP
7. Transfer Learning for NLP 1: Transformer Model
8. Transfer Learning for NLP 2: BERT & GPT
9. Speech Recognition



AI Mastery Course

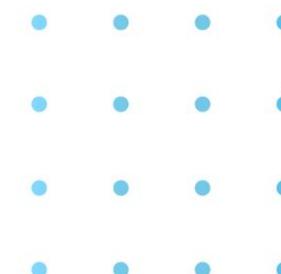


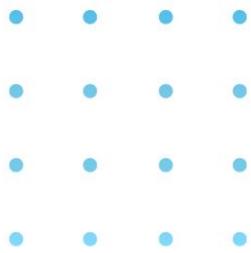
Module 10

Natural Language Processing (NLP)

Section 1

Introduction to Natural Language Processing

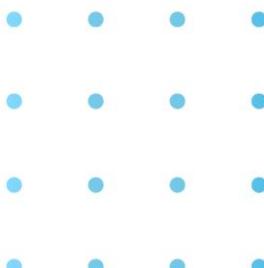




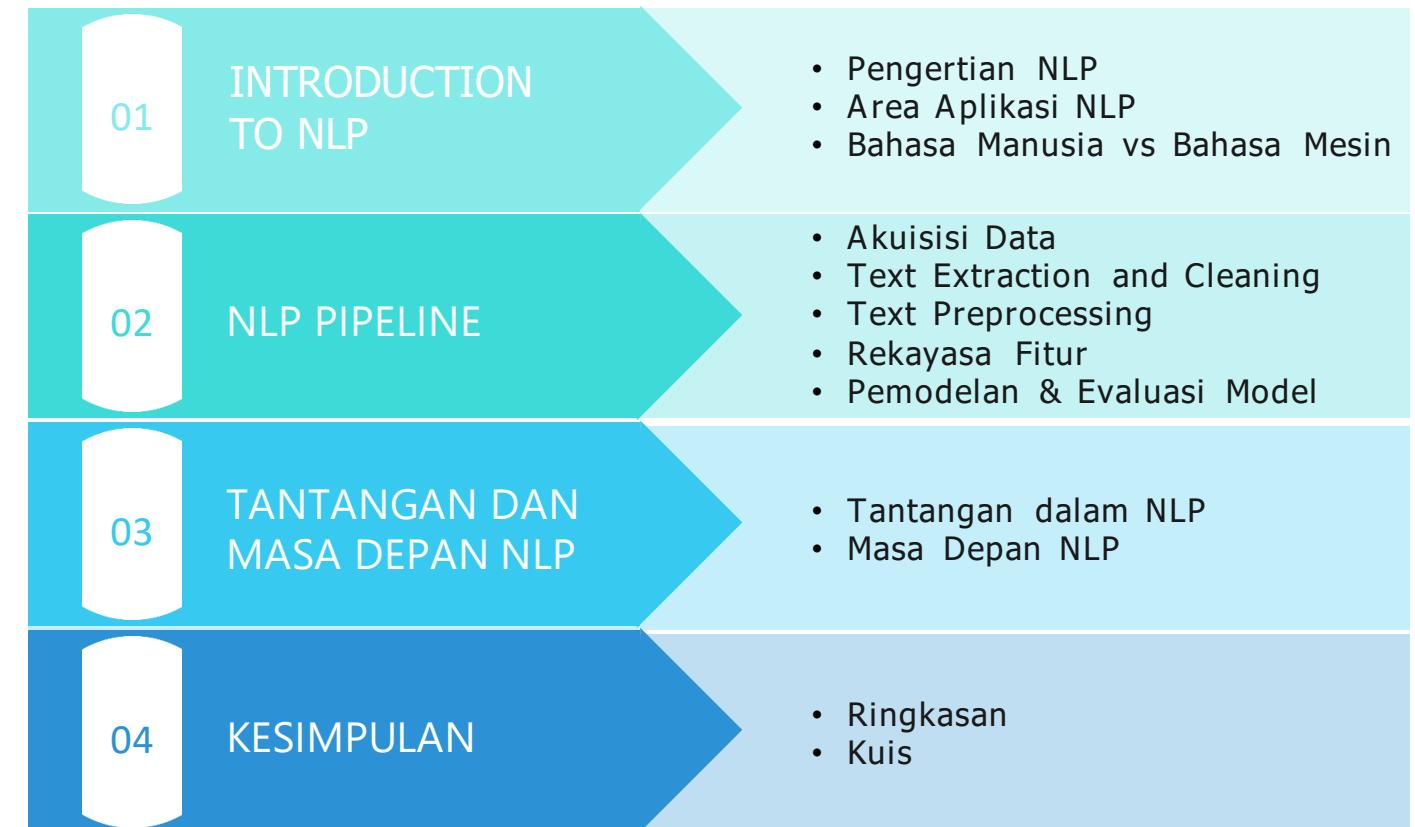
Learning Objectives

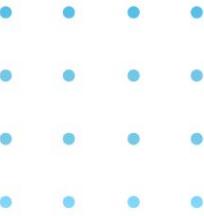
Di akhir modul ini, Anda akan mendapatkan:

- Memahami NLP dan evolusinya.
- Memahami area aplikasi NLP dan penerapannya.
- Memahami NLP pipeline secara umum dan teknik-tekniknya
- Memahami tantangan dan masa depan NLP



Agenda



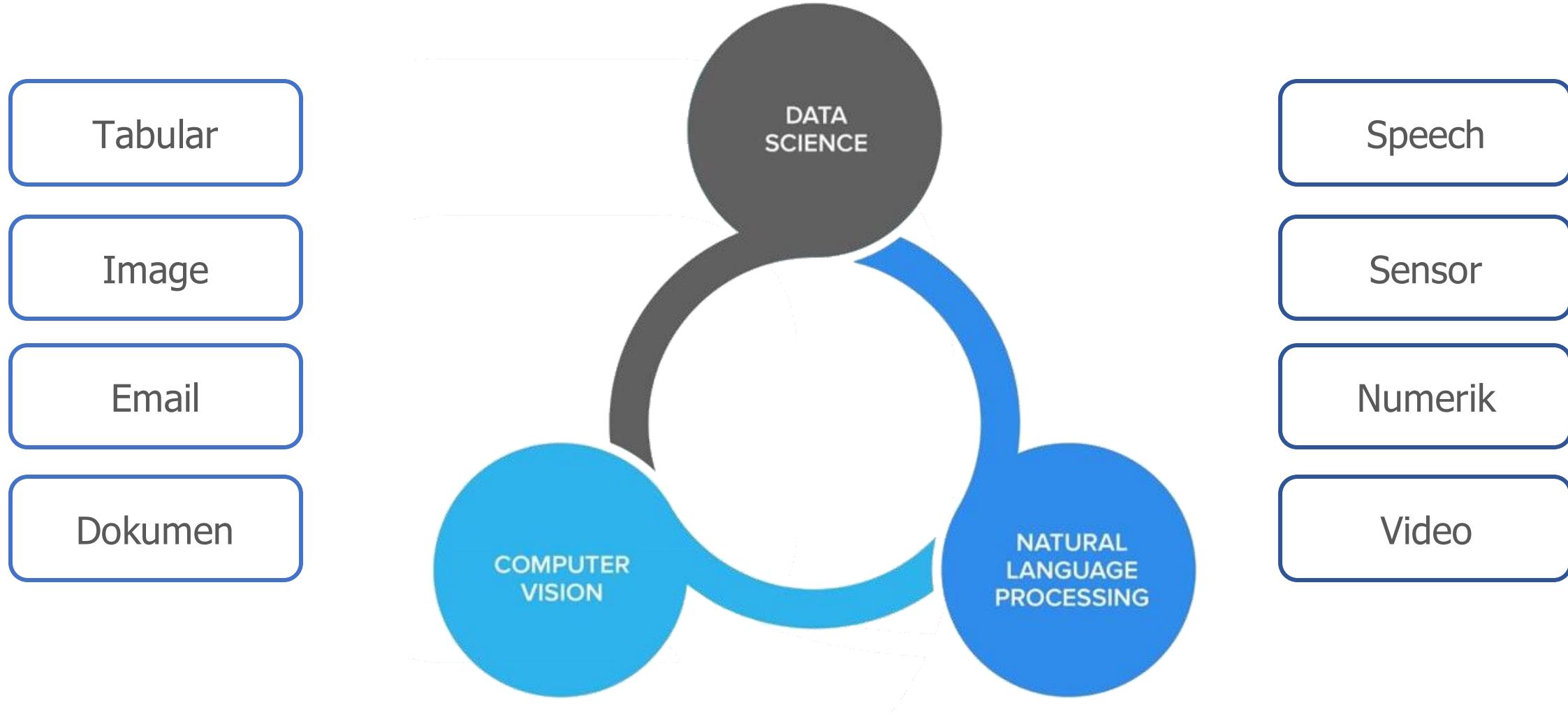


01 INTRODUCTION TO NLP

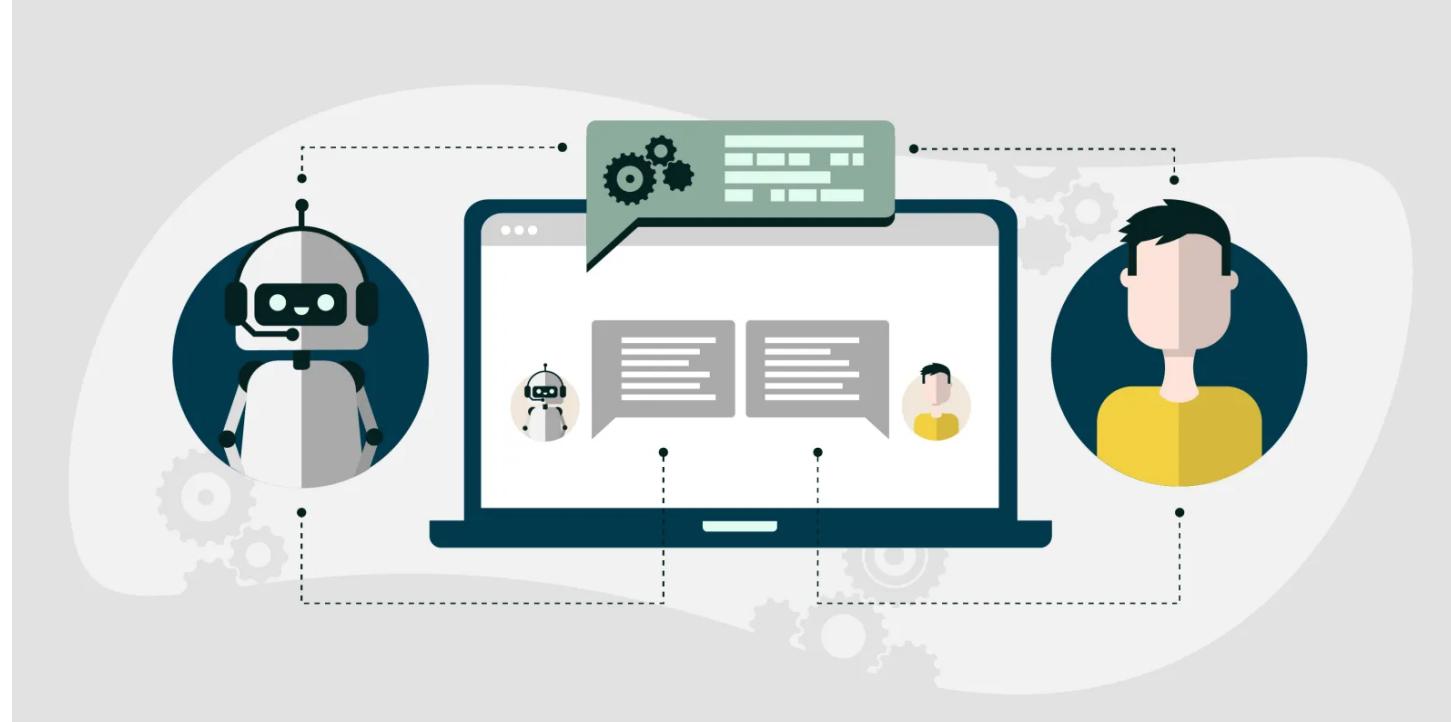
- Pengertian NLP
- Area aplikasi NLP
- Bahasa manusia vs mesin



AI Domains: Recap

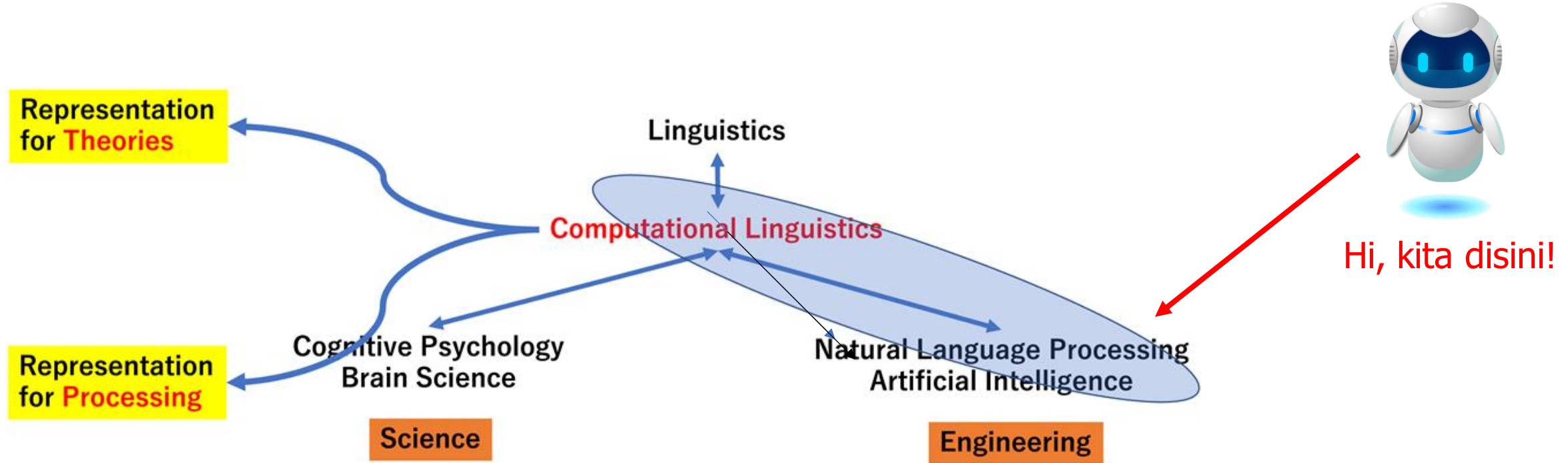


Jadi, apa itu NLP?



NLP cabang dari kecerdasan buatan yang berhubungan dengan interaksi antara komputer dan manusia menggunakan bahasa alami.

NLP dan Linguistik

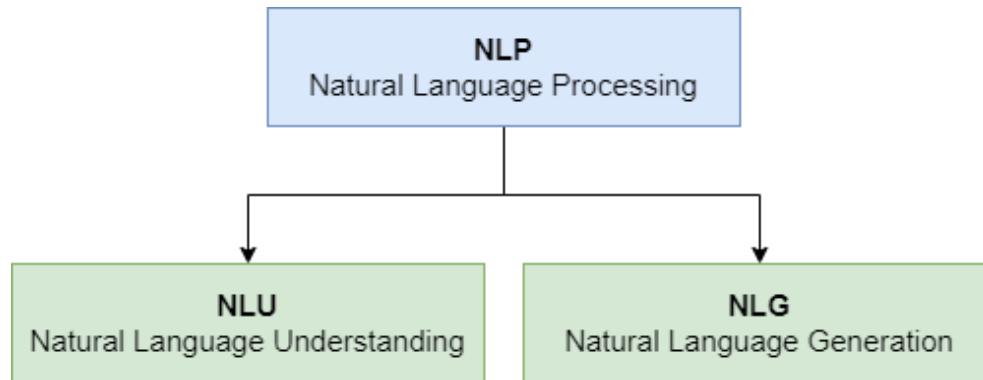


Baca selanjutnya:

J. Tsuji, "Natural Language Processing and Computational Linguistics," *Computational Linguistics*, pp. 1–21, Dec. 2021.

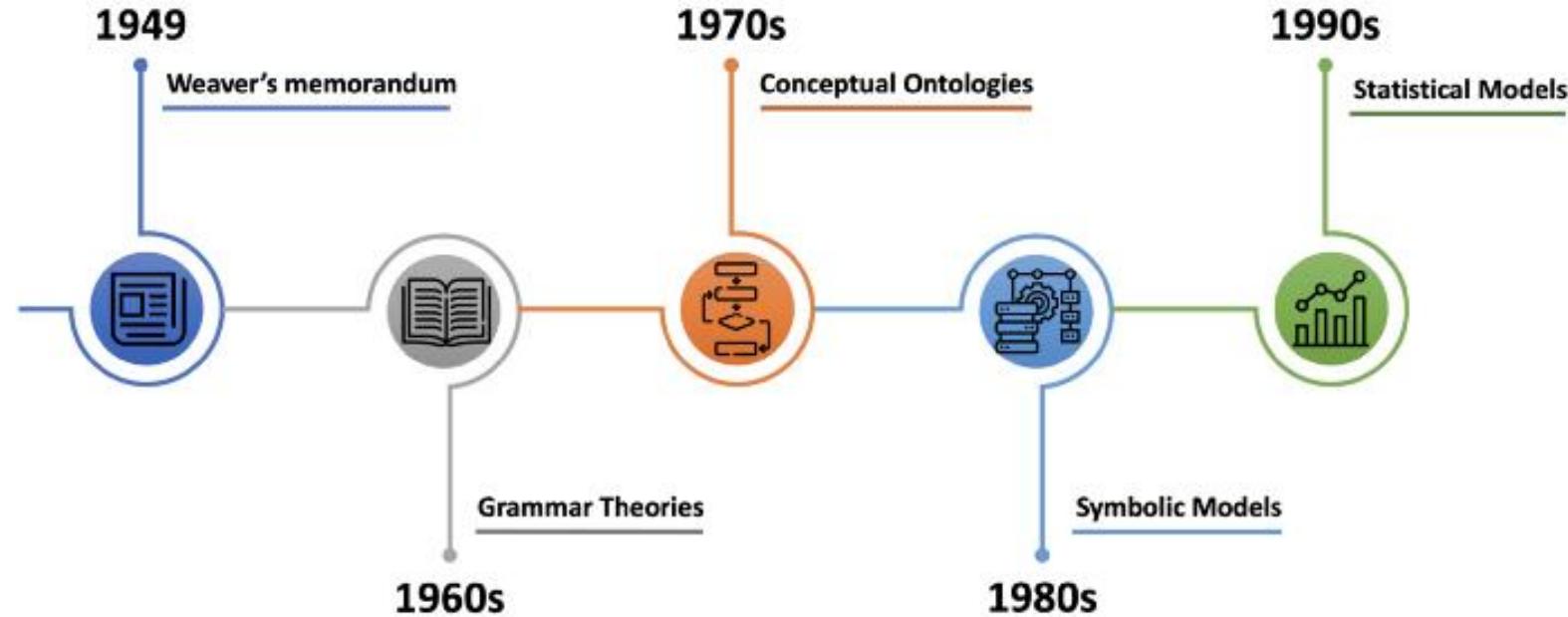
https://doi.org/10.1162/coli_a_00420

NLP = NLU + NLG

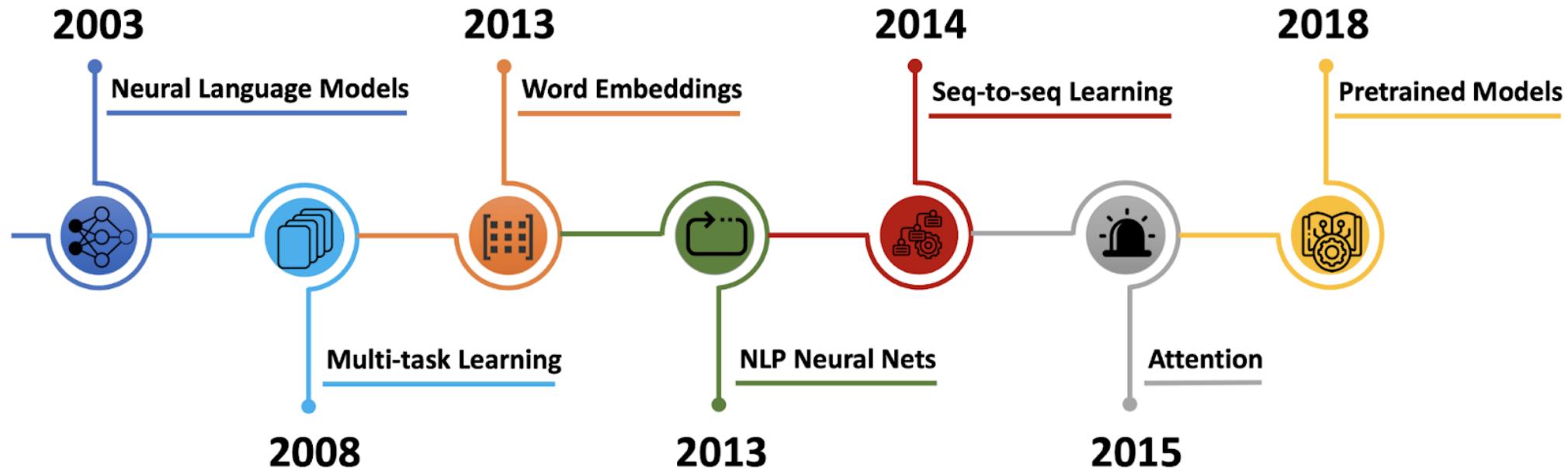


- NLP adalah kemampuan komputer memahami bahasa manusia dalam bentuk tertulis (teks) dan verbal (ucapan).
- NLU adalah bagian NLP yang menggunakan analisis sintaksis dan semantik untuk menentukan makna kalimat (membaca).
- NLG adalah bagian NLP untuk menghasilkan respons bahasa manusia berdasarkan beberapa input (menulis).

Sejarah NLP – Era Klasik



Sejarah NLP – Era Deep Learning



Baca selanjutnya:

P. Johri, S. K. Khatri, A. T. Al-Taani, M. Sabharwal, S. Suvanov, and A. Kumar, "Natural Language Processing: History, Evolution, Application, and Future Work," *Lecture Notes in Networks and Systems*, pp. 365–375, 2021, doi: 10.1007/978-981-15-9712-1_31.

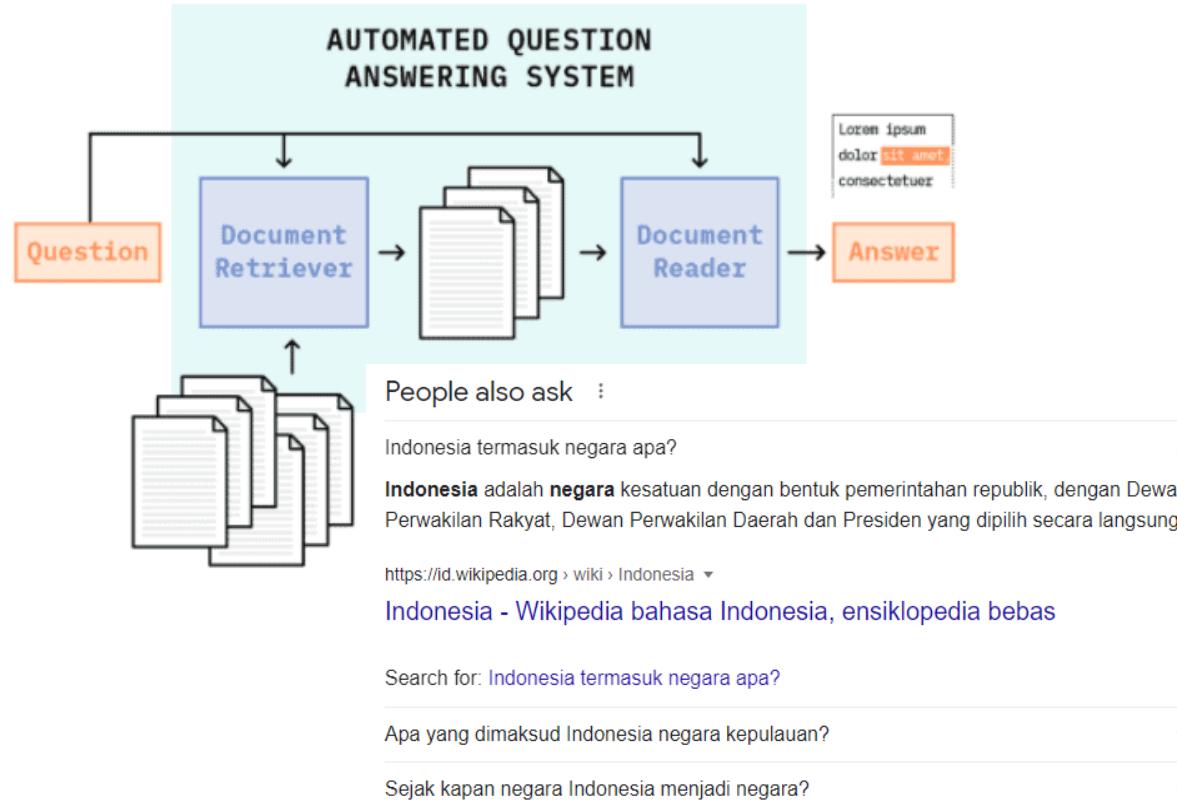
https://doi.org/10.1007/978-981-15-9712-1_31

Untuk apa NLP digunakan?



Tanpa kita sadari, **Natural Language Processing** adalah kekuatan pendorong di balik aplikasi yang umum kita gunakan saat ini.

Question Answering System (QAS)



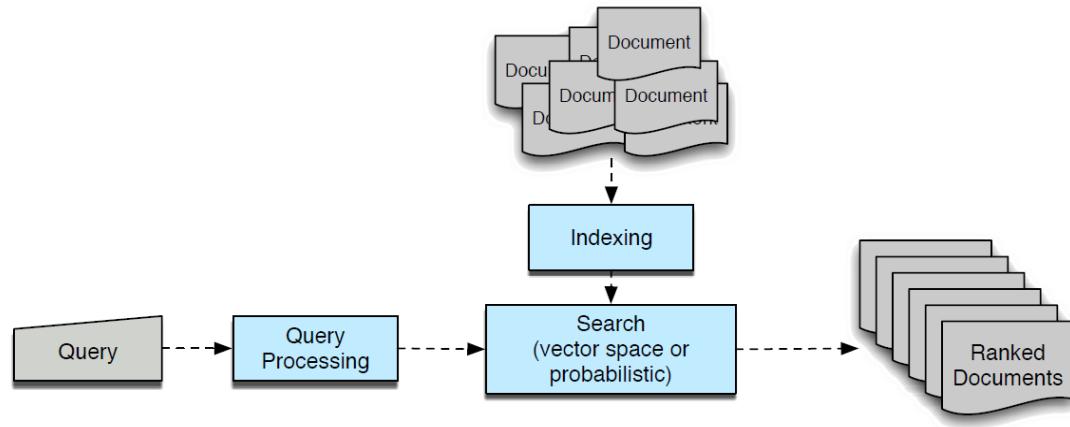
Kemampuan komputer untuk menjawab pertanyaan yang diberikan oleh pengguna.

Baca selanjutnya:

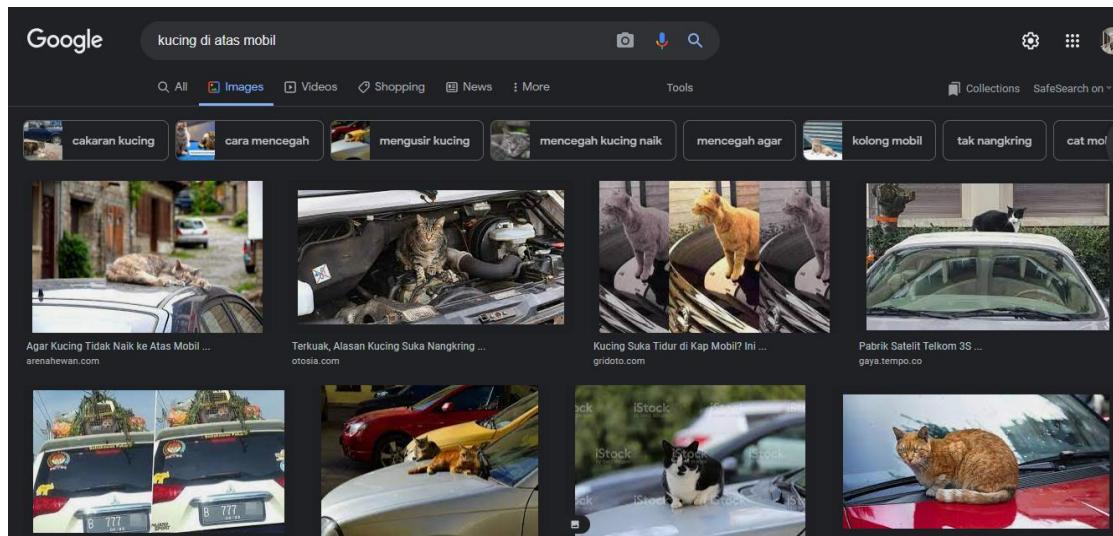
A. Bouziane, D. Bouchiha, N. Doumi, and M. Malki, "Question Answering Systems: Survey and Trends," *Procedia Computer Science*, vol. 73, pp. 366–375, 2015.

<https://doi.org/10.1016/j.procs.2015.12.005>

Information Retrieval



Kemampuan komputer mencari konten yang relevan berdasarkan *query* (kata kunci) yang diberikan pengguna.



Baca selanjutnya:

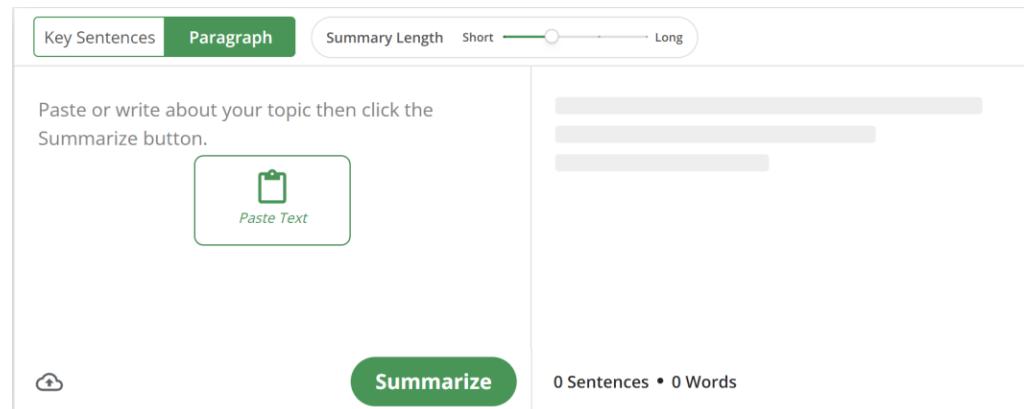
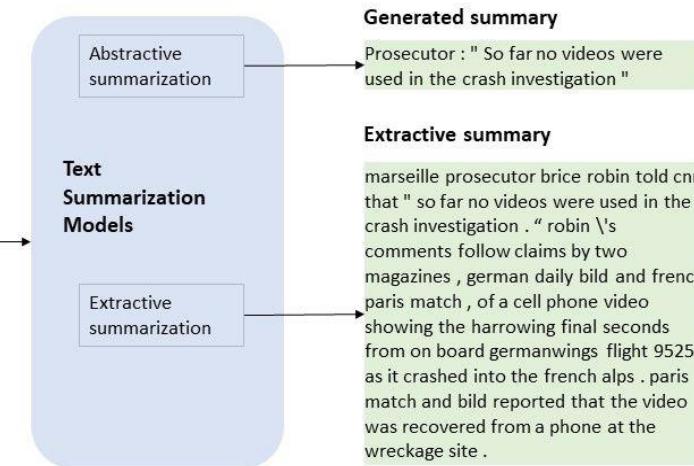
H. K. Azad and A. Deepak, "Query expansion techniques for information retrieval: A survey," *Information Processing & Management*, vol. 56, no. 5, pp. 1698–1735, Sep. 2019.

<https://doi.org/10.1016/j.ipm.2019.05.009>

Text Summarization

Input Article

Marseille, France (CNN) The French prosecutor leading an investigation into the crash of Germanwings Flight 9525 insisted Wednesday that he was not aware of any video footage from on board the plane. Marseille prosecutor Brice Robin told CNN that " so far no videos were used in the crash investigation ." He added, " A person who has such a video needs to immediately give it to the investigators ." Robin's comments follow claims by two magazines, German daily Bild and French Paris Match, of a cell phone video showing the harrowing final seconds from on board Germanwings Flight 9525 as it crashed into the French Alps . All 150 on board were killed. Paris Match and Bild reported that the video was recovered from a phone at the wreckage site. ...



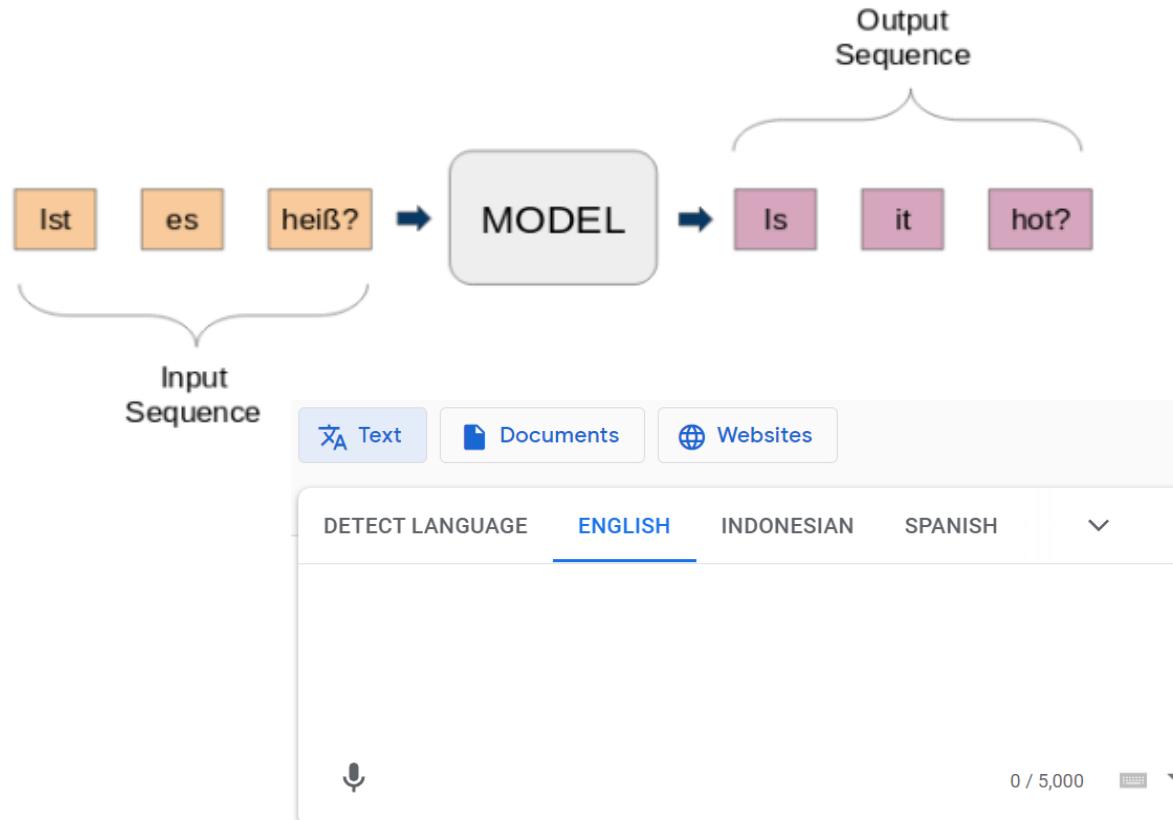
Kemampuan komputer untuk meringkas konten dengan mencari informasi paling penting atau relevan dalam konten asli.

Baca selanjutnya:

W. S. El-Kassas, C. R. Salama, A. A. Rafea, and H. K. Mohamed, "Automatic text summarization: A comprehensive survey," *Expert Systems with Applications*, vol. 165, p. 113679, Mar. 2021

<https://doi.org/10.1016/j.eswa.2020.113679>

Machine Translation



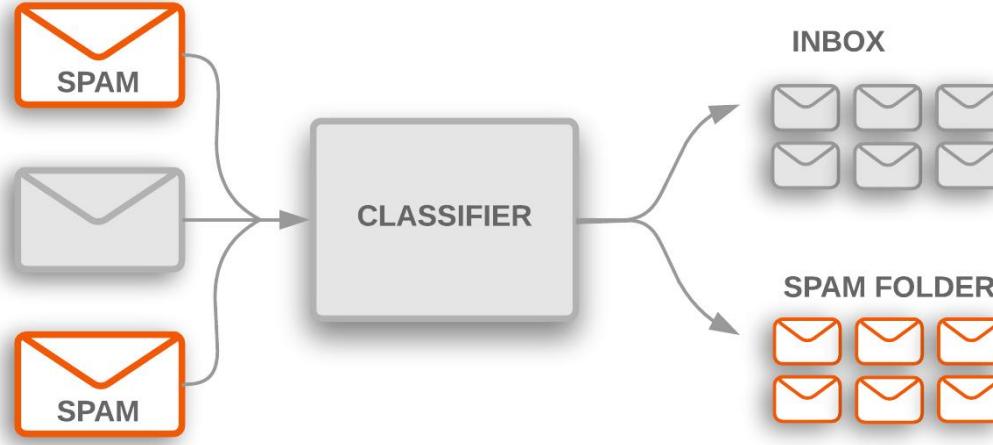
Kemampuan komputer untuk menerjemahkan konten dari satu bahasa ke bahasa lain secara otomatis.

Baca selanjutnya:

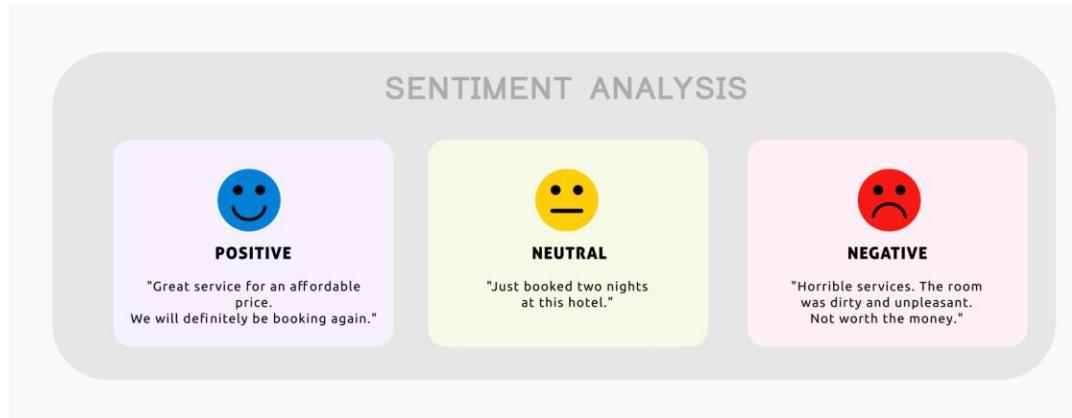
H. Wang, H. Wu, Z. He, L. Huang, and K. Ward Church, "Progress in Machine Translation," *Engineering*, Jul. 2021.

<https://doi.org/10.1016/j.eng.2021.03.023>

Text Classification



Kemampuan komputer untuk mengkategorikan konten ke satu atau lebih kategori secara otomatis.

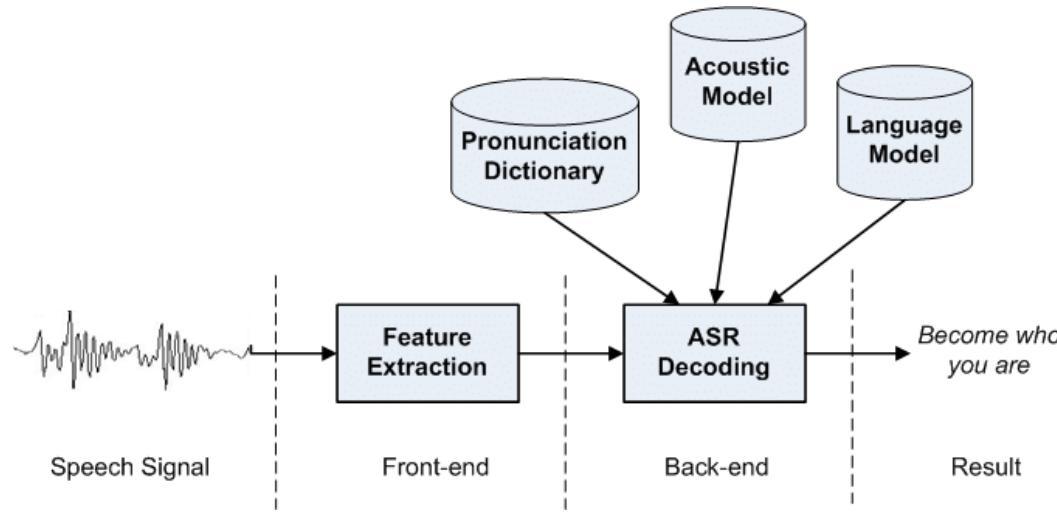


Baca selanjutnya:

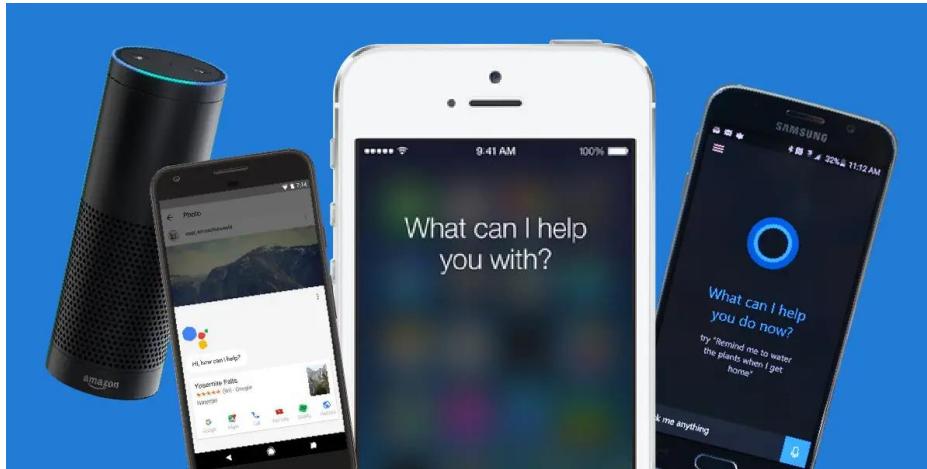
B. Altinel and M. C. Ganiz, "Semantic text classification: A survey of past and recent advances," *Information Processing & Management*, vol. 54, no. 6, pp. 1129–1153, Nov. 2018.

<https://doi.org/10.1016/j.ipm.2018.08.001>

Speech Recognition



Kemampuan komputer untuk mengenali dan menerjemahkan bahasa lisan ke dalam teks secara otomatis.

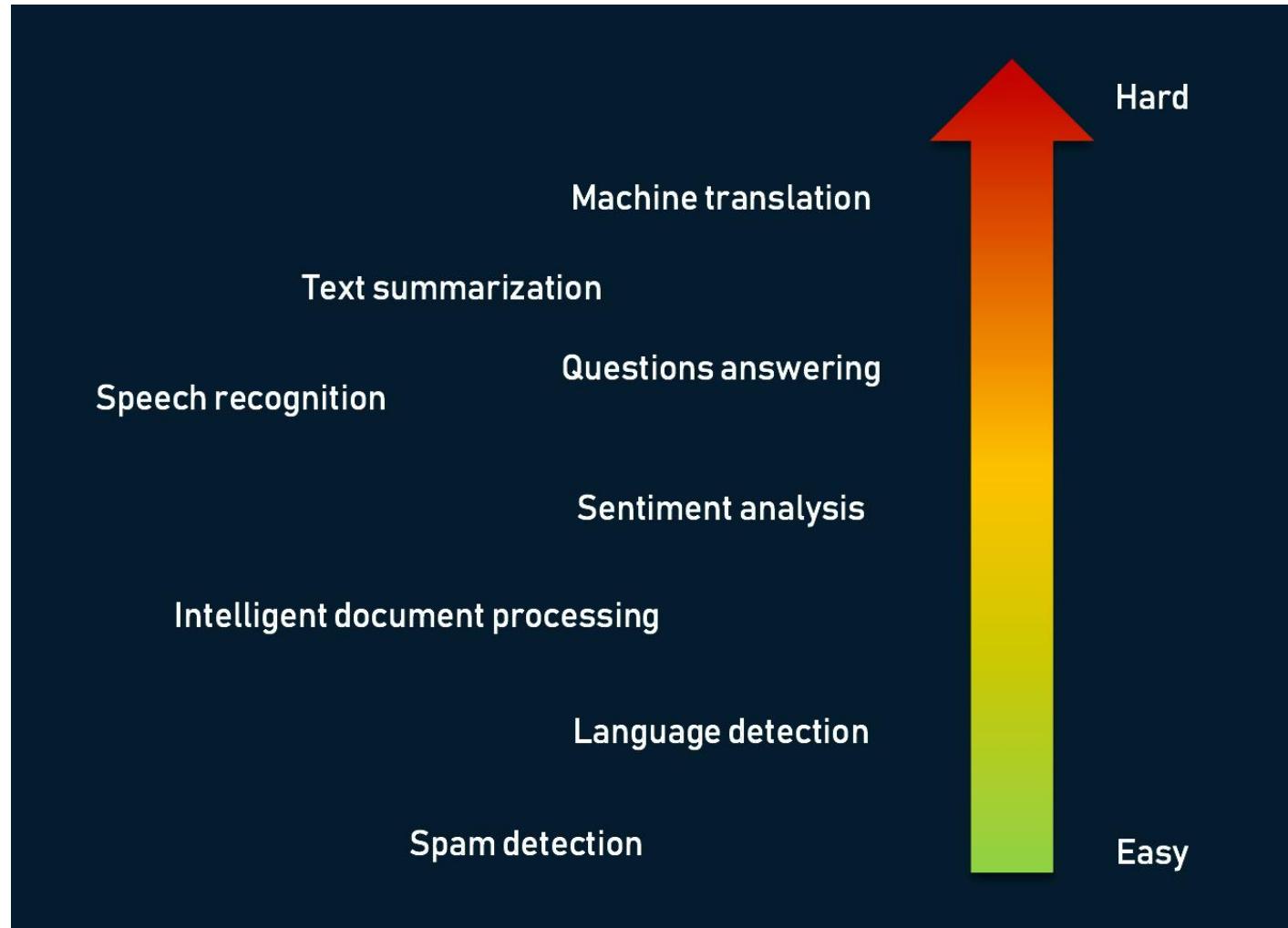


Baca selanjutnya:

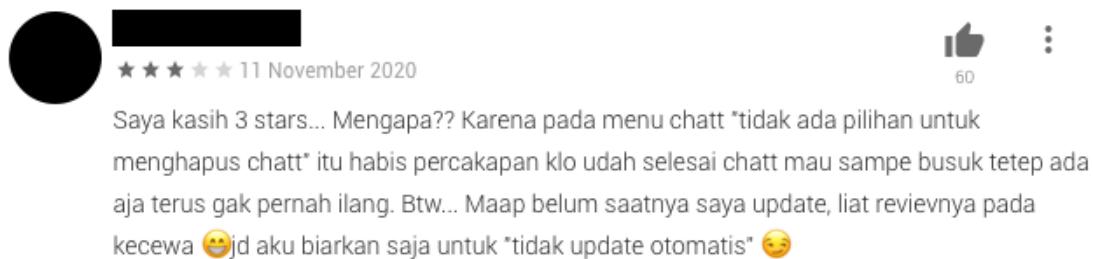
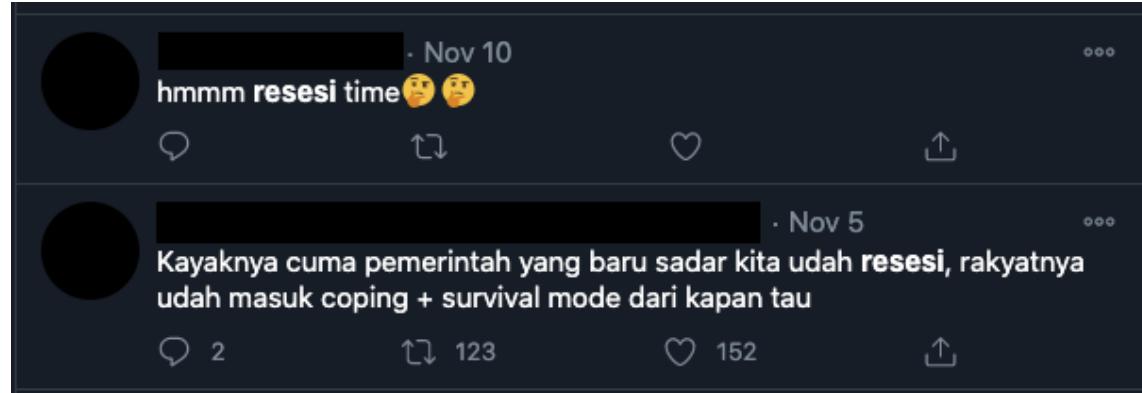
A. P. Singh, R. Nath, and S. Kumar, "A Survey: Speech Recognition Approaches and Techniques," *2018 5th IEEE Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON)*, Nov. 2018.

[10.1109/UPCON.2018.8596954](https://doi.org/10.1109/UPCON.2018.8596954)

Mana yang lebih sulit?



Tapi, bagaimana mesin membaca?



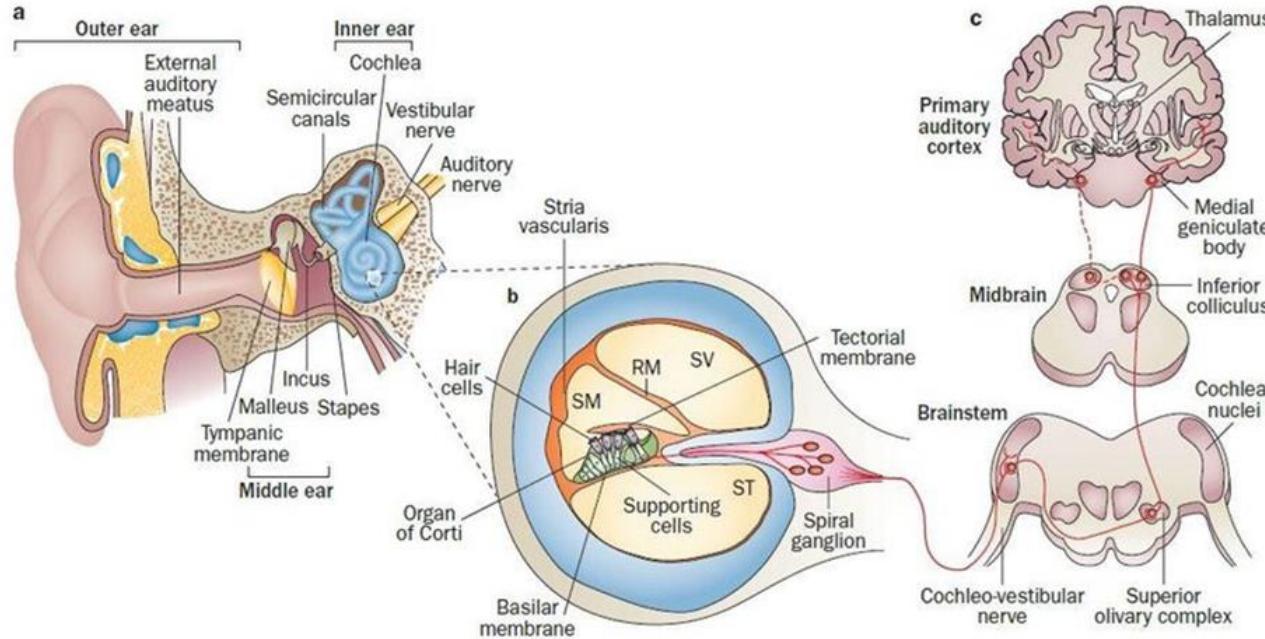
Kepada Yth,
Dinas Kependudukan dan Pencatatan Sipil

selamat siang, saya ingin bertanya apabila ktp sementara hilang dan urgent ingin mengambil ktp karena kepentingan mendadak apa bisa menggunakan kk?

ULASAN PALING MEMBANTU

A card for a helpful review. It shows a user profile with a black circle, a 5-star rating, and the text "4 bulan lalu". Below it is a snippet of the review: "Sudah di coba online meeting dengan hasil memuaskan di audio dan video.". At the bottom, there's a question "Apakah ulasan ini membantu?", a "like" button with "3", and a "Laporkan" button.

Bahasa manusia vs mesin



script.py

```
1 facebook = "Facebook's rating is"
2 fb_rating = 3.5
3
4 fb_rating_str = str(3.5)
5 fb = facebook + ' ' + fb_rating_str
6
7 print(fb)
```

Susunan Kata dan Artinya

Semantik

Kajian yang mencakup arti dan makna kata sesungguhnya dalam satuan kalimat.

Sintaksis

Kajian yang mencakup seluk-beluk tata bahasa (grammar & cara penulisan) dalam satuan kalimat.

Analogi dengan bahasa pemrograman :

Sintaks berbeda, semantik sama

(5): $2+3 = 3+2$

Sintaks sama, semantik berbeda

(1 dan 1.5)

Makna dari Sebuah Kata

Wajahnya menjadi **merah** setelah dia mengetahui bahwa dia mengambil tas yang salah.

Andi membeli mobil warna **merah**.

Wajahnya menjadi **merah** setelah meminum obatnya.

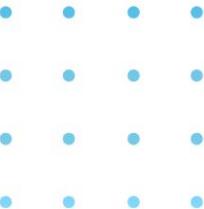
Hm, bagaimana dengan komputer?



Memahami Bahasa manusia adalah hal yang **sulit** untuk dilakukan oleh **mesin**.

Lalu, bagaimana membuat komputer mengerti Bahasa kita?

Ya! Kita perlu mengubah teks menjadi angka!

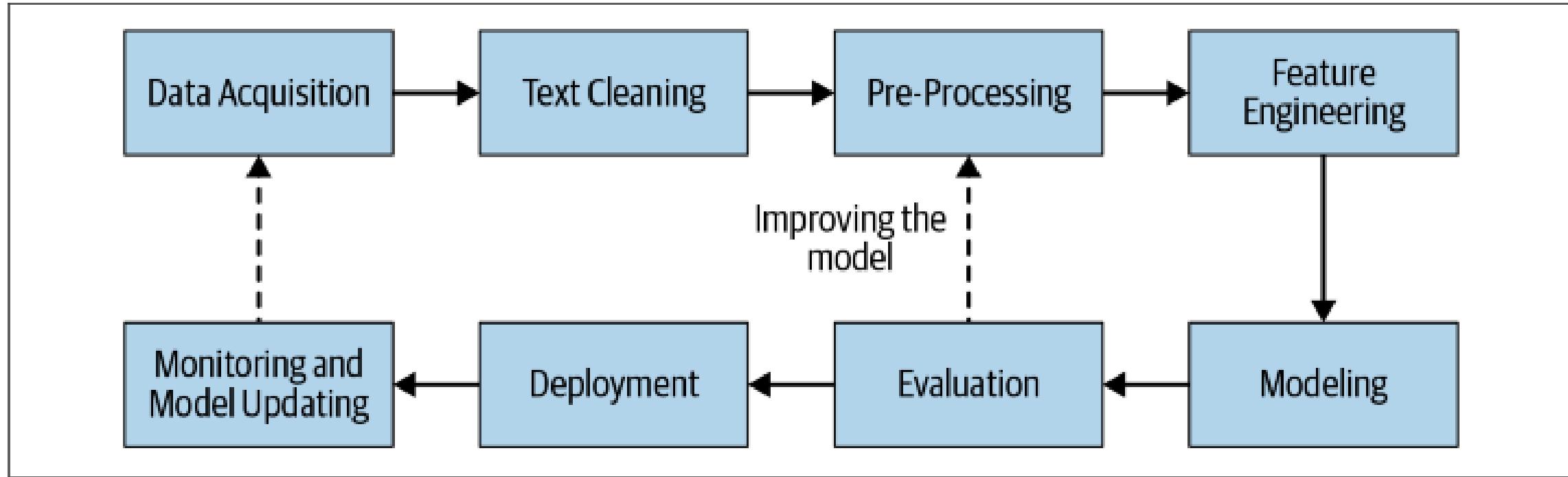


02 NLP PIPELINE

- Akuisisi Data
- Text Extraction and Cleaning
- Text Preprocessing
- Rekayasa Fitur
- Pemodelan & Evaluasi Model



NLP Pipeline



01 Akuisisi Data

Public Dataset

Menggunakan data publik yang sudah tersedia melalui berbagai sumber seperti Hugging Face, Google Dataset Search, Kaggle, dll.

Web Scraping

Mengumpulkan data secara langsung dari halaman web menggunakan beautiful soup, selenium, scrapy.

Product Intervention

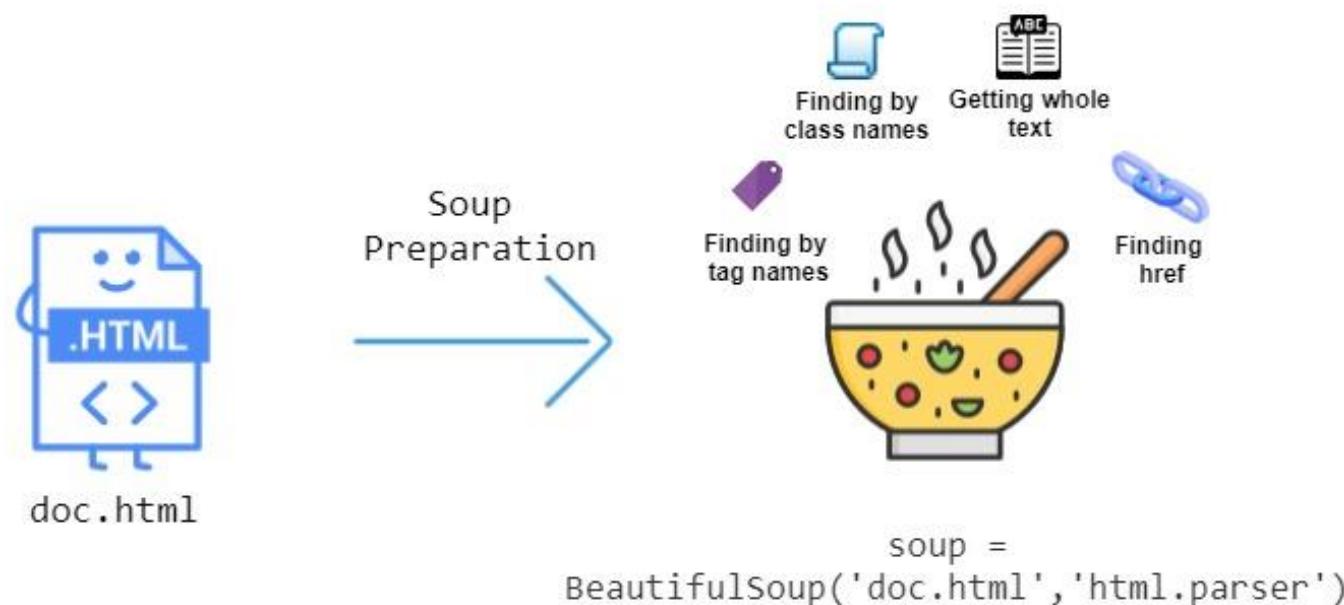
Mengumpulkan data dari produk sendiri atau dari produk yang sudah ada. Misalnya

Data Augmentation

Menghasilkan lebih banyak data dari kumpulan dataset yang ada

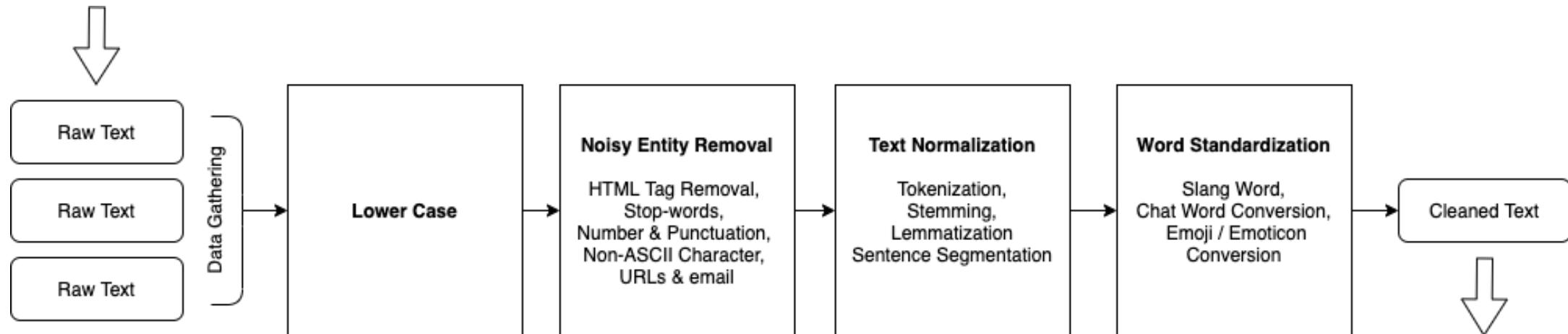
02 Text Extraction & Cleaning

Langkah ini mengacu pada proses **mengekstrak teks mentah** dari data input dengan **menghapus semua informasi non-teksual**, seperti metadata, tag HTML dan mengonversi teks ke format yang diperlukan. Langkah ini bersifat opsional, tergantung pada format data yang tersedia.



03 Text Preprocessing

Langkah ini mengacu pada proses **menyeleksi teks agar lebih terstruktur** dengan melalui serangkaian tahapan. Tidak semua tahapan text preprocessing harus dilakukan, tergantung pada tugas dan domain yang akan dikerjakan.



03 Text Processing

Lower Case & Remove Whitespace

Tahapan paling sederhana, mudah, dan efektif pada text preprocessing.

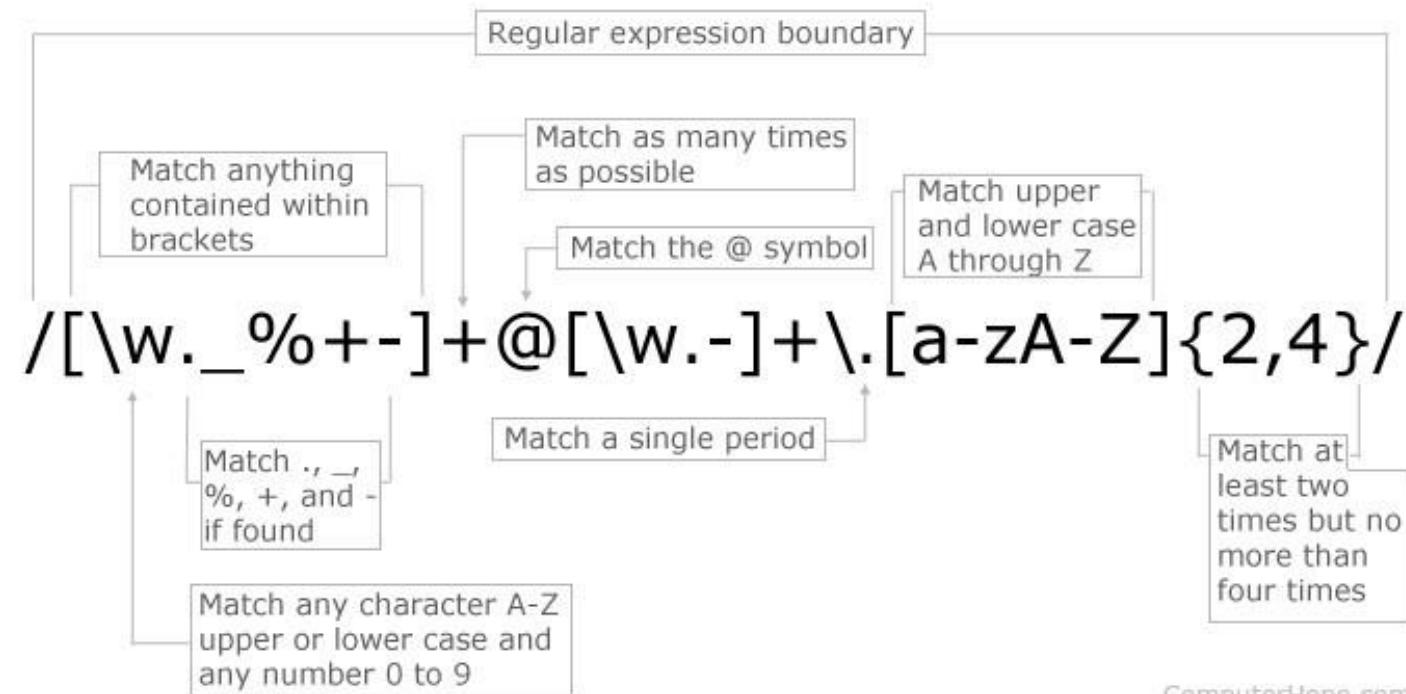
Indonesia ≠ INDONESIA ≠ indonesia

```
1 def lowercase(text):
2     text = text.lower().strip()
3     return text
```

03 Text Processing

Regular Expression (ReGex)

RegEx adalah string teks (urutan karakter) untuk membuat pola yang membantu mencocokkan, menemukan, dan mengelola teks.



03 Text Processing

Remove URLs & Email

```
● ● ●  
1 import re  
2  
3 def remove_url(text):  
4     text = re.sub(r'https?://\S+|www\.\S+', '', text)  
5     text = re.sub(r'pic.twitter.com\S+', '', text)  
6     return text  
7  
8 def remove_email(text):  
9     text = re.sub('\S*@\S*\s?', '', text)  
10    return text
```

03 Text Processing

Remove Numbers & Punctuations

```
● ● ●  
1 import re  
2  
3 def remove_numbers(text):  
4     text = re.sub('[-+]?[0-9]+', '', text)  
5     return text  
6  
7 def remove_punctuation(text):  
8     text = re.sub(r'^\w\s', '', text)  
9     return text
```

03 Text Processing

Remove Emoji & Emoticon

```
1 import re
2
3 def remove_emoji(text):
4     emoji_pattern = re.compile("["
5         u"\U0001F600-\U0001F64F"
6         u"\U0001F300-\U0001F5FF"
7         u"\U0001F680-\U0001F6FF"
8         u"\U0001F1E0-\U0001F1FF"
9         u"\U00002500-\U00002BEF"
10    "+", flags=re.UNICODE)
11    text = emoji_pattern.sub(r'', text)
12    return text
```



```
1 from emoji_unicode import UNICODE_EMO, EMOTICONS
2
3 def remove_emoticon(text):
4     emoticon_pattern = re.compile(u'(' + u'|'.join(k for k in EMOTICONS) + u')')
5     text = emoticon_pattern.sub(r'', text)
6     return text
```

03 Text Processing

Emoji & Emoticon Conversion

" → grinning_face atau :-) → happy_face_smiley"

```
● ● ●

1 import re
2 from emoji_unicode import UNICODE_EMO, EMOTICONS
3
4 def convert_emoji(text):
5     for emoji in UNICODE_EMO:
6         text = text.replace(emoji, '_'.join(UNICODE_EMO[emoji].replace(' ','_').replace(':','_').split()))
7
8 return text
```

```
● ● ●

1 import re
2 from emoji_unicode import UNICODE_EMO, EMOTICONS
3
4 def convert_emoticon(text):
5     for emoticon in EMOTICONS:
6         text = re.sub(u'('+emoticon+')', '_'.join(EMOTICONS[emoticon].replace(' ','_').split()), text)
7
8 return text
```

03 Text Processing

Slang Word Normalization

Tahapan mengubah kata slang menjadi kata baku.
“gmn, gims → bagaimana, jwb → jawab, gue, gw → saya”



```
1 def normalize_word(text):
2     normal_word_path = pd.read_csv('../Data/key_norm.csv')
3
4     text = ' '.join([normal_word_path[normal_word_path['singkat'] == word]['hasil'].values[0]
5     if (normal_word_path['singkat'] == word).any() else word for word in text.split()])
6
7     return text
```

03 Text Processing

Stemming & Lemmatization

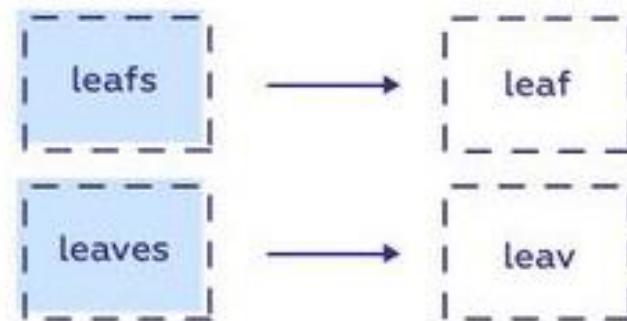
Stemming

Tahapan menghilangkan prefix dan suffix menjadikan kata ke bentuk dasar.

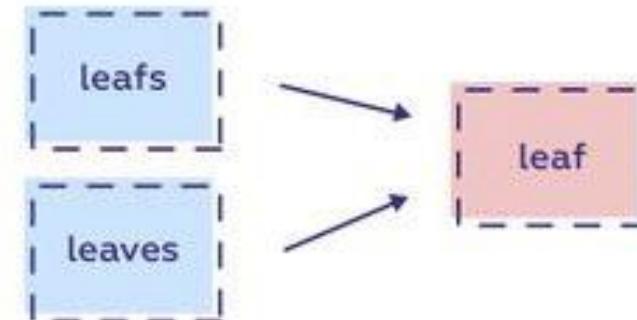
Lemmatization

Tahapan mengubah kata ke bentuk dasar dengan memperhatikan pengetahuan linguistik.

Stemming



Lemmatization



03 Text Processing

Stemming Indonesian

Pada Bahasa Indonesia kita dapat menggunakan library Sastrawi
“mendengarkan, dengarkan, didengarkan → dengar”

| Prefix | Dissallowed Suffixes |
|--------|----------------------|
| be- | -i |
| di- | -an |
| ke- | -i, -kan |
| me- | -an |
| te- | -i, -kan |
| se- | -an |

```

1 from Sastrawi.Stemmer.StemmerFactory import
   StemmerFactory
2
3 def stemming(text):
4     factory = StemmerFactory()
5     stemmer = factory.create_stemmer()
6     text = stemmer.stem(text)
7     return text

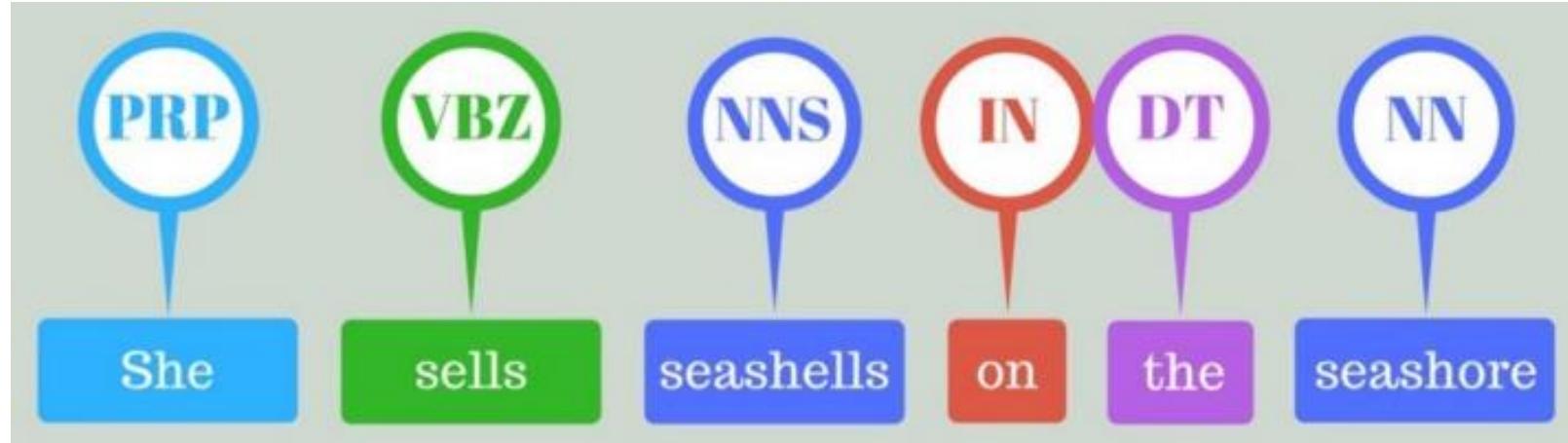
```

03 Text Processing

Part of Speech Tagging

POS Tagging

Memberi label pada kata-kata dalam suatu teks menurut jenis katanya.

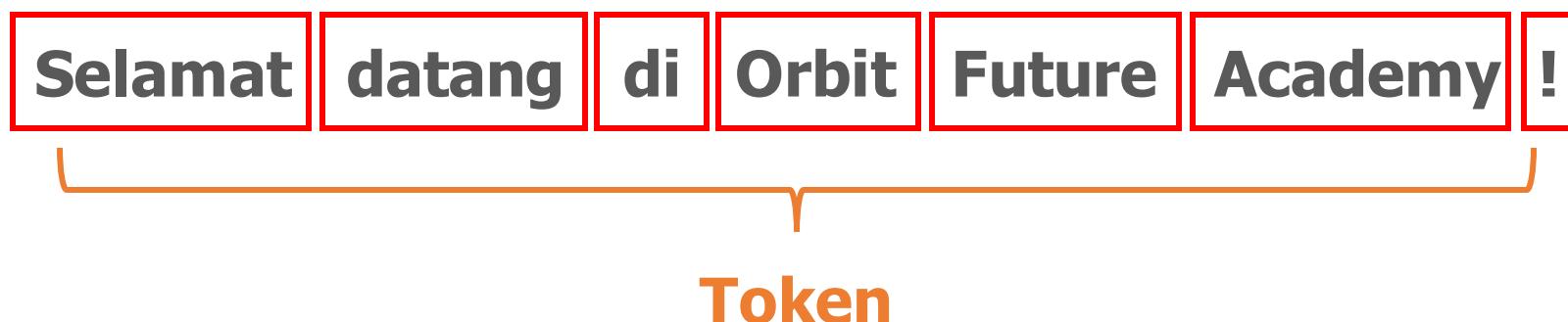


03 Text Preprocessing

Tokenization

Tahapan **pemisahan teks** (kata atau kalimat) menjadi **potongan** yang disebut **token**. Kalimat, kata-kata, angka, simbol, tanda baca, dan entitas penting lainnya dapat dianggap sebagai token.

“Selamat datang di Orbit Future Academy !”



03 Text Preprocessing

Tokenization

```
● ● ●  
1 import nltk  
2 from nltk.tokenize import sent_tokenize, word_tokenize  
3  
4 def word_tokenize(text):  
5     words = nltk.word_tokenize(text)  
6     return words  
7  
8 def sent_tokenize(text):  
9     sent = nltk.sent_tokenize(text)  
10    return sent
```

sent_tokenize()

Memisahkan kalimat pada suatu paragraf.

word_tokenize()

Memisahkan kata pada suatu kalimat.

03 Text Processing

Stop words Removal

Tahapan **menghapus kata-kata berinformasi rendah** (noise). Karakteristik utama pemilihan stop word adalah kata yang mempunyai **frekuensi kemunculan yang tinggi** misalnya kata penghubung seperti 'dan', 'atau', 'tapi'.

Tidak ada aturan pasti dalam menentukan stop word yang akan digunakan. Penentuan stop word bisa disesuaikan dengan domain atau tugas yang sedang diselesaikan.

03 Text Processing

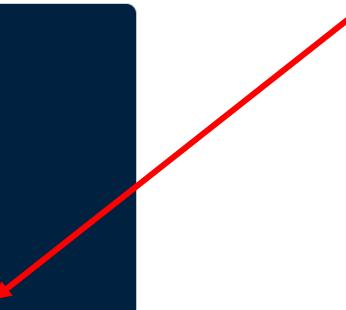
Stop words Removal



```
1 from nltk.corpus import stopwords
2
3 def stopwords_removal(words):
4     stopword = stopwords.words('indonesian')
5     more_stopword = ['daring', 'online', 'covid-19']
6     stop_factory = stopword + more_stopword
7
8     clean_words = []
9     for word in words:
10         if word not in stop_factory:
11             clean_words.append(word)
12     words = clean_words
13     return words
```



Tambahkan stop word berdasarkan tugas yang akan diselesaikan



```
1 from Sastrawi.StopWordRemover.StopWordRemoverFactory
2 import StopWordRemoverFactory
3
4 def stopwords_removal_sastrawi(text):
5     factory = StopWordRemoverFactory()
6     stopword = factory.create_stop_word_remover()
7     text = stopword.remove(text)
8     return text
```

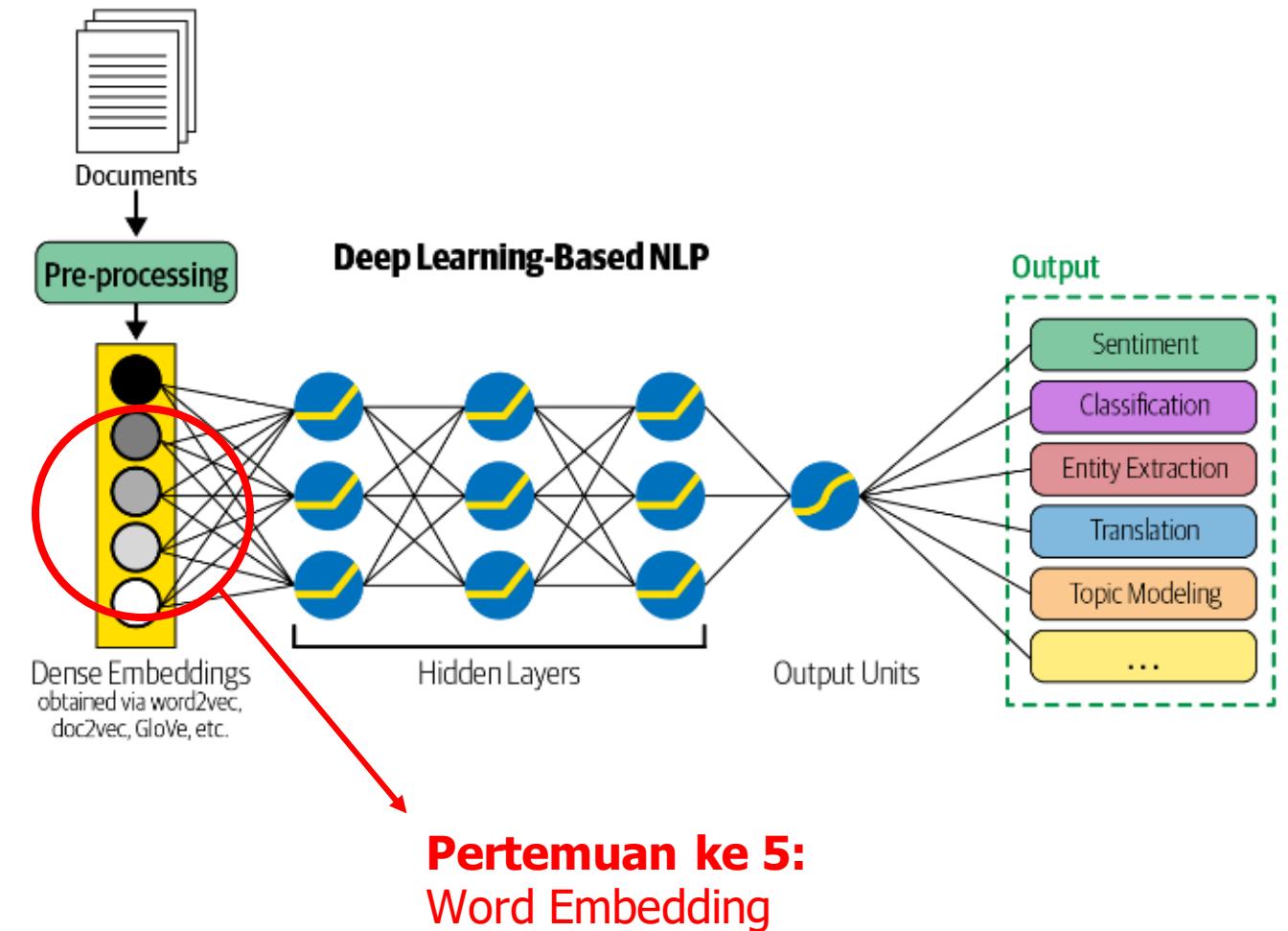
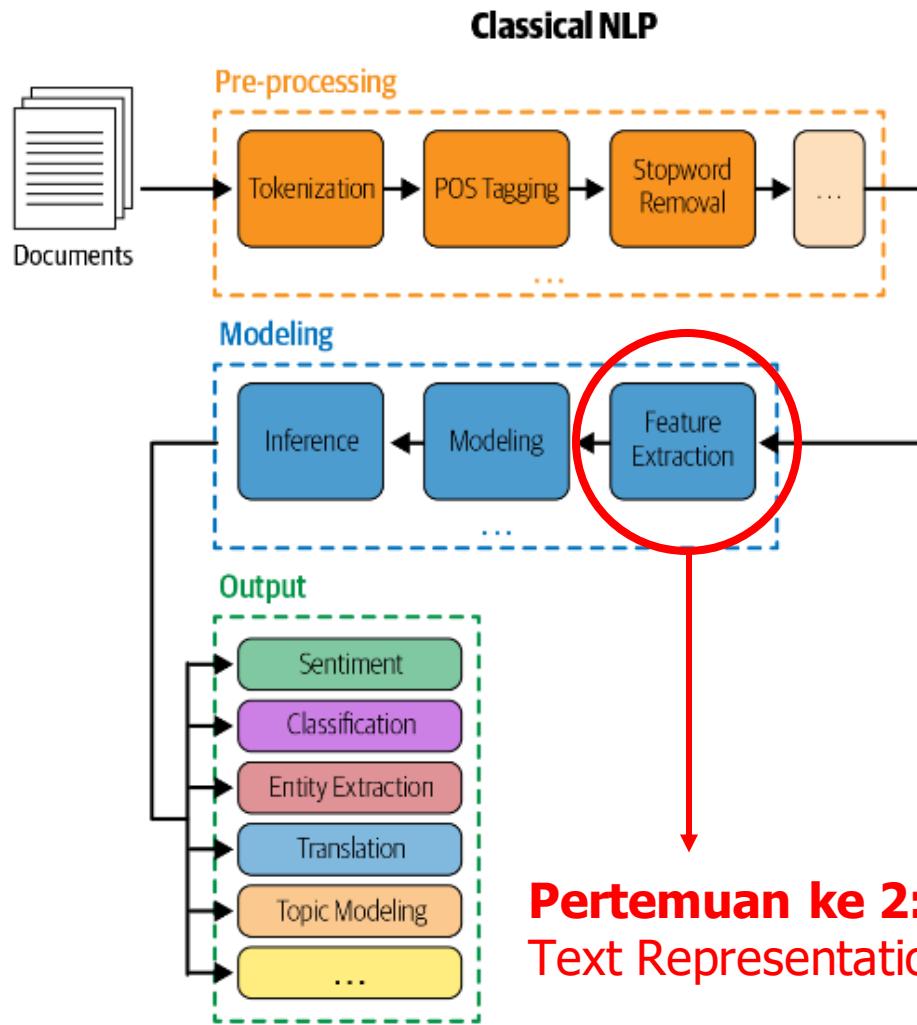
04 Rekayasa Fitur

Rekayasa fitur adalah **seni**. Pada NLP, dapat disebut '**representasi teks**'.

Langkah ini bertujuan untuk menangkap **karakteristik teks** menjadi **vektor numerik** yang dapat dipahami oleh **algoritma**.

| | cat | mat | on | sat | the |
|---------------|-----|-----|----|-----|-----|
| the => | 0 | 0 | 0 | 0 | 1 |
| cat => | 1 | 0 | 0 | 0 | 0 |
| sat => | 0 | 0 | 0 | 1 | 0 |
| ... | ... | | | | |

04 Rekayasa Fitur Machine Learning vs Deep Learning



05 Pemodelan

Mulai latih dan kembangkan model NLP. Saat ini, banyak model dikembangkan berbasis deep learning. Sehingga, membutuhkan sumber daya komputasi yang tinggi.



06 Evaluasi Model

Dalam setiap pengembangan model AI, langkah kuncinya adalah mengukur seberapa 'bagus' model yang dibuat.

Keberhasilan pada langkah ini meliputi:

1. Menggunakan metrik yang tepat untuk evaluasi model;
2. Mengikuti proses evaluasi yang tepat;

Baca selanjutnya:

M.-A. Clinciu, A. Eshghi, and H. Hastie, "A Study of Automatic Metrics for the Evaluation of Natural Language Explanations," 2387. Accessed: Mar. 20, 2022.

<https://aclanthology.org/2021.eacl-main.202.pdf>



06 Evaluasi Model

| Metric | Aplikasi |
|--|--|
| Accuracy | Banyak digunakan untuk klasifikasi teks, misalnya sentiment analysis |
| Precision | Banyak digunakan jika kesalahan di kelas positif lebih penting daripada kesalahan di kelas negatif, misalnya prediksi penyakit. |
| Recall | Banyak digunakan jika pengambilan hasil kelas positif lebih penting, misalnya pencarian e-commerce atau tugas information-retrieval lainnya. |
| Mean Reciprocal Rank (MRR) | Banyak digunakan untuk tugas information-retrieval. |
| Bilingual Evaluation Understudy (BLUE) | Banyak digunakan untuk tugas machine translation dan chatbot. |

Dan banyak lagi ...



03

TANTANGAN DAN MASA DEPAN NLP

- Tantangan NLP
- Masa depan NLP

Tantangan

- Sarkasme
- Ambiguitas kata dan kalimat
- Kesalahan dalam teks atau ucapan
- Penggunaan bahasa gaul
- Konteks bahasa
- Bahasa khusus sesuai domain (kesehatan, sains, sejarah)
- Bahasa dengan sumber daya yang sedikit
- Dataset Bahasa Indonesia yang terbatas

Masa Depan NLP

- Pengenalan emosi
- Bahasa berkembang secara dinamis, analisis sintaks seperti struktur bahasa akan terus ada.
- Kebutuhan AI untuk faham mengenai makna dan logika dari teks dan suara akan terus dibutuhkan

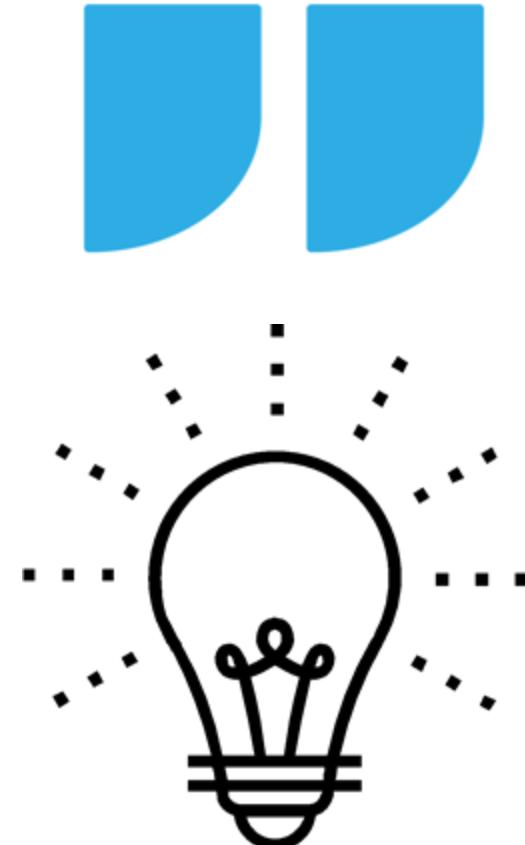


04 KESIMPULAN

- Ringkasan
- Kuis

Ringkasan

1. NLP adalah cabang keilmuan dari kecerdasan buatan yang mempelajari interaksi antara komputer dan manusia menggunakan bahasa alami.
2. NLP terdiri dari NLU (kemampuan membaca) dan NLG (kemampuan menulis).
3. Area aplikasi NLP: QAS, information retrieval, text summarization, text classification, machine translation.
4. NLP pipeline merujuk pada langkah-langkah yang dilakukan untuk membangun sistem berbasis NLP. Secara umum terdiri dari: akuisisi data, text cleaning, text processing, rekayasa fitur, pemodelan, evaluasi model, deployment dan monitoring.
5. Tantangan utama NLP adalah sarkasme, kesalahan Bahasa (typo), penggunaan Bahasa gaul, dan ketersediaan Bahasa.
6. Masa depan NLP meliputi pengenalan emosi, kebutuhan pemahaman Bahasa akan terus dibutuhkan pada semua bidang kehidupan manusia.



Kuis

Pertanyaan

Penerapan NLP di dunia nyata adalah

- A. Object Detection
- B. Self-Driving cars
- C. Sentiment Analysis
- D. Object Segmentation



Kuis

Pertanyaan

Penerapan NLP di dunia nyata adalah

- A. Object Detection
- B. Self-Driving cars
- C. Sentiment Analysis**
- D. Object Segmentation



Jawaban: C



TERIMA KASIH

Orbit Future Academy

PT Orbit Ventura Indonesia
Center of Excellence (Jakarta Selatan)
Gedung Veteran RI, Lt.15
Unit Z15-002, Plaza Semanggi
Jl. Jenderal Sudirman Kav.50, Jakarta
12930, Indonesia

- Jakarta Selatan/Pusat
- Jakarta Barat/BSD
- Kota Bandung
- Kab. Bandung
- Jawa Barat

Hubungi Kami

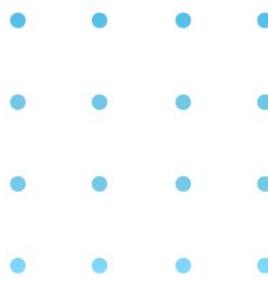
Director of Sales & Partnership
ira@orbitventura.com
+62 858-9187-7388

Social Media

-  [Orbit Future Academy](#)
-  [@OrbitFutureAcademyIn1](#)
-  [OrbitFutureAcademy](#)
-  [Orbit Future Academy](#)



AI Mastery Course

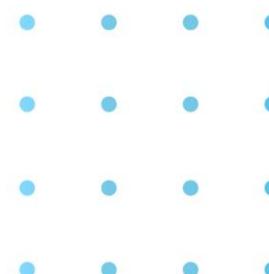


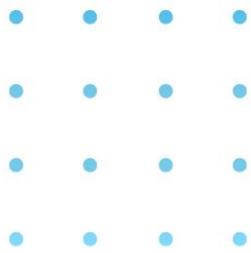
Module 10

Natural Language Processing (NLP)

Section 2

Text Representation

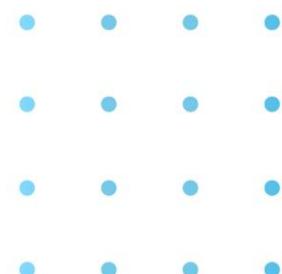




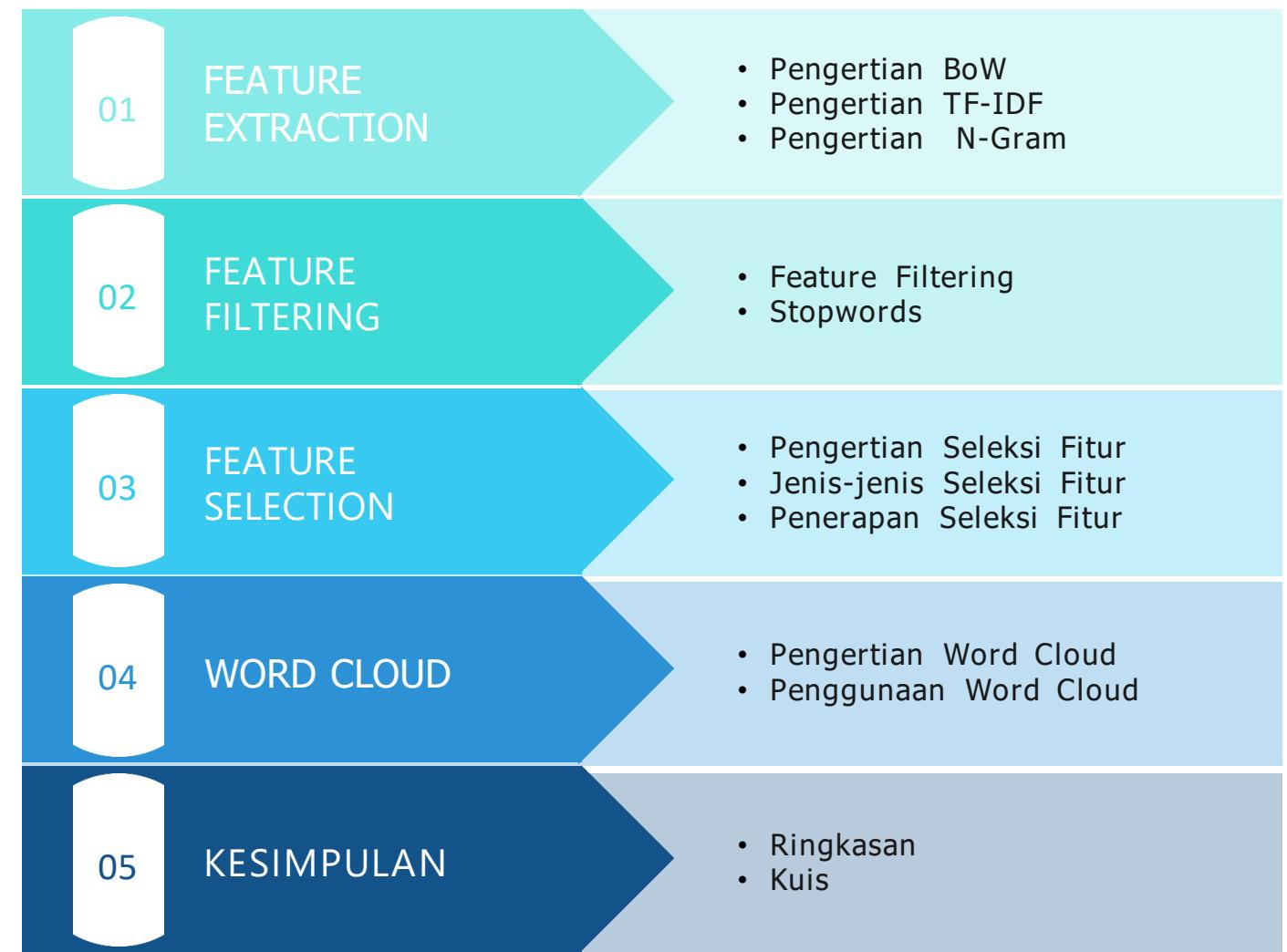
Learning Objectives

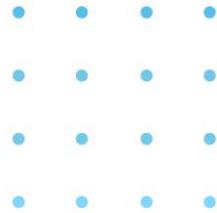
Di akhir modul ini, Anda akan mendapatkan:

- Memahami proses feature extraction menggunakan BoW/ count vectorizer dan TF-IDF.
- Mengetahui jenis-jenis N-gram dan pengunaanya
- Mampu menggunakan feature filtering
- Mengetahui jenis-jenis feature selection
- Mampu menggunakan dan memanfaatkan word cloud



Agenda





01 FEATURE EXTRACTION

- BoW/Count Vectorizer
- TF-IDF
- N-Gram



Feature Extraction (Pembobotan kata)

Pembobotan kata adalah proses mengubah kata menjadi bentuk vektor agar bisa diproses oleh algoritme machine learning. Ada dua pembobotan kata yang umum digunakan, yaitu:

1. Bag of Word (BoW) / Count Vectorizer
2. Term Frequency-Inverse Document Frequency (TF-IDF)

Saya suka belajar di Orbit Future Academy

| Kata | Frekuensi |
|---------|-----------|
| Saya | 1 |
| Suka | 1 |
| Belajar | 0 |
| di | 2 |
| Orbit | 3 |
| Future | 1 |
| Academy | 0 |

Contoh Teks to vektor

1. Bag of Word (BoW) /Count Vectorizer

BoW merupakan sebuah model yang mempelajari kosa kata dari seluruh dokumen, kemudian memodelkan tiap dokumen dengan menghitung jumlah setiap kata yang muncul [1]

Contoh teks

Saya suka belajar di Orbit Future Academy

Saya senang belajar AI

Bersama Orbit saya dan teman saya bisa pintar

[1] W. T. H. Putri and R. Hendrowati, "Penggalian Teks Dengan Model Bag of Words Terhadap Data Twitter," *J. Muara Sains, Teknol. Kedokteran, dan Ilmu Kesehat.*, vol. 2, no. 1, pp. 129–138, 2018.

1. Bag of Word (BoW) /Count Vectorizer

Berdasarkan data teks sebelumnya, dibentuk sebuah list.

Saya suka belajar di Orbit Future Academy

Saya senang belajar AI

Bersama Orbit saya dan teman saya bisa pintar

“saya”, “suka”, “belajar”, “di”, “orbit”, “future”, “academy”

“saya”, “senang”, “belajar”, “ai”

“bersama”, “orbit”, “saya”, “dan”, “teman”, “saya”, “bisa”, “pintar”

1. Bag of Word (BoW) /Count Vectorizer

Kemudian, setiap kata dihitung frekuensinya dan dipetakan kembali ke dalam tabel.

“saya”, “suka”, “belajar”, “di”, “orbit”, “future”, “academy”

“saya”, “senang”, “belajar”, “ai”

“bersama”, “orbit”, “saya”, “dan”, “teman”, “saya”, “bisa”, “pintar”

| | saya | suka | belajar | di | orbit | future | academy | senang | ai | bersama | dan | teman | bisa | pintar |
|----|------|------|---------|----|-------|--------|---------|--------|----|---------|-----|-------|------|--------|
| T1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| T2 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| T3 | 2 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |

Tabel di atas menggambarkan fitur yang mengandung frekuensi kata dari tiap kalimat dalam sebuah dokumen.

2. TF-IDF



TF - IDF (Term Frequency — Inverse Document Frequency)

Merupakan ukuran penilaian yang umumnya digunakan dalam pencarian informasi (IR) dan peringkasan teks.

Skor TF-IDF menunjukkan seberapa penting atau relevan suatu istilah (kata) dalam dokumen tertentu.

2. TF-IDF



Jika kata tertentu muncul beberapa kali dalam sebuah dokumen, maka kata tersebut mungkin memiliki tingkat kepentingan yang lebih tinggi daripada kata lain yang muncul lebih sedikit (TF).

Pada saat yang sama, jika kata tertentu muncul berkali-kali dalam sebuah dokumen, tetapi juga muncul berkali-kali di beberapa dokumen lain, maka mungkin kata itu sering muncul, jadi kita tidak dapat menganggapnya penting (IDF).

2. Notasi TF-IDF

$$TF_{(t,d)} = \frac{f_{t,d}}{\sum f_{t,d}} \dots \dots \dots \quad 1.1$$

Dengan:

$f_{t,d}$ = Frekuensi setiap kata (t) yang muncul di dalam kalimat (d)

$\sum t, d$ = Total keseluruhan kata (t) yang terdapat di dalam kalimat (d)

2. Notasi TF-IDF

Persamaan (1.2) digunakan untuk mencari nilai IDF. Penambahan angka 1 untuk menghindari angka 0 jika $df(t)$ tidak ditemukan pada *corpus*.

Dengan:

$|D|$ = Jumlah dokumen / kalimat yang ada dalam koleksi

$df(t)$ = Jumlah dokumen / kalimat di mana muncul kata (t)

2. Notasi TF-IDF

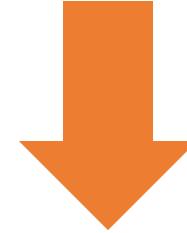
Sehingga Persamaan (1.3) digunakan untuk menghitung nilai TF-IDF.

2. Contoh TF-IDF

Buruh tolak omnibus law omnibus law jangan disahkan

Omnibus law meningkatkan ekonomi

Omnibus law jangan disahkan



“Buruh”, “tolak”, “omnibus”, “law”, “omnibus”, “law”, “jangan”, “disahkan”

“Omnibus”, “law”, “meningkatkan”, “ekonomi”

“Omnibus”, “law”, “jangan”, “disahkan”

2. Contoh TF-IDF

Perhitungan Nilai TF-IDF

| Fitur/Kata | TF | | | DF | IDF | TF.IDF | | |
|--------------|-------|------|------|----|-------|--------|-------|-------|
| | T1 | T2 | T3 | | | T1 | T2 | T3 |
| Buruh | 0,125 | 0 | 0 | 1 | 1,386 | 0,173 | 0 | 0 |
| Tolak | 0,125 | 0 | 0 | 1 | 1,386 | 0,173 | 0 | 0 |
| Omnibus | 0,25 | 0,25 | 0,25 | 3 | 0,693 | 0,173 | 0,173 | 0,173 |
| Law | 0,25 | 0,25 | 0,25 | 3 | 0,693 | 0,173 | 0,173 | 0,173 |
| Meningkatkan | 0 | 0,25 | 0 | 1 | 1,386 | 0 | 0,346 | 0 |
| Ekonomi | 0 | 0,25 | 0 | 1 | 1,386 | 0 | 0,346 | 0 |
| Jangan | 0,125 | 0 | 0,25 | 2 | 0,916 | 0,114 | 0 | 0,229 |
| Disahkan | 0,125 | 0 | 0,25 | 2 | 0,916 | 0,114 | 0 | 0,229 |

2. Contoh TF-IDF

Vektor kata dalam kalimat menggunakan TF-IDF

| No | Fitur 1 | Fitur 2 | Fitur 3 | Fitur 4 | Fitur 5 | Fitur 6 | Fitur 7 | Fitur 8 |
|----|---------|---------|---------|---------|--------------|---------|---------|----------|
| | Buruh | Tolak | Omnibus | Law | Meningkatkan | Ekonomi | Jangan | Disahkan |
| 1 | 0,173 | 0,173 | 0,173 | 0,173 | 0 | 0 | 0,114 | 0,114 |
| 2 | 0 | 0 | 0,173 | 0,173 | 0,346 | 0,346 | 0 | 0 |
| 3 | 0 | 0 | 0,173 | 0,173 | 0 | 0 | 0,229 | 0,229 |

3. N-Gram

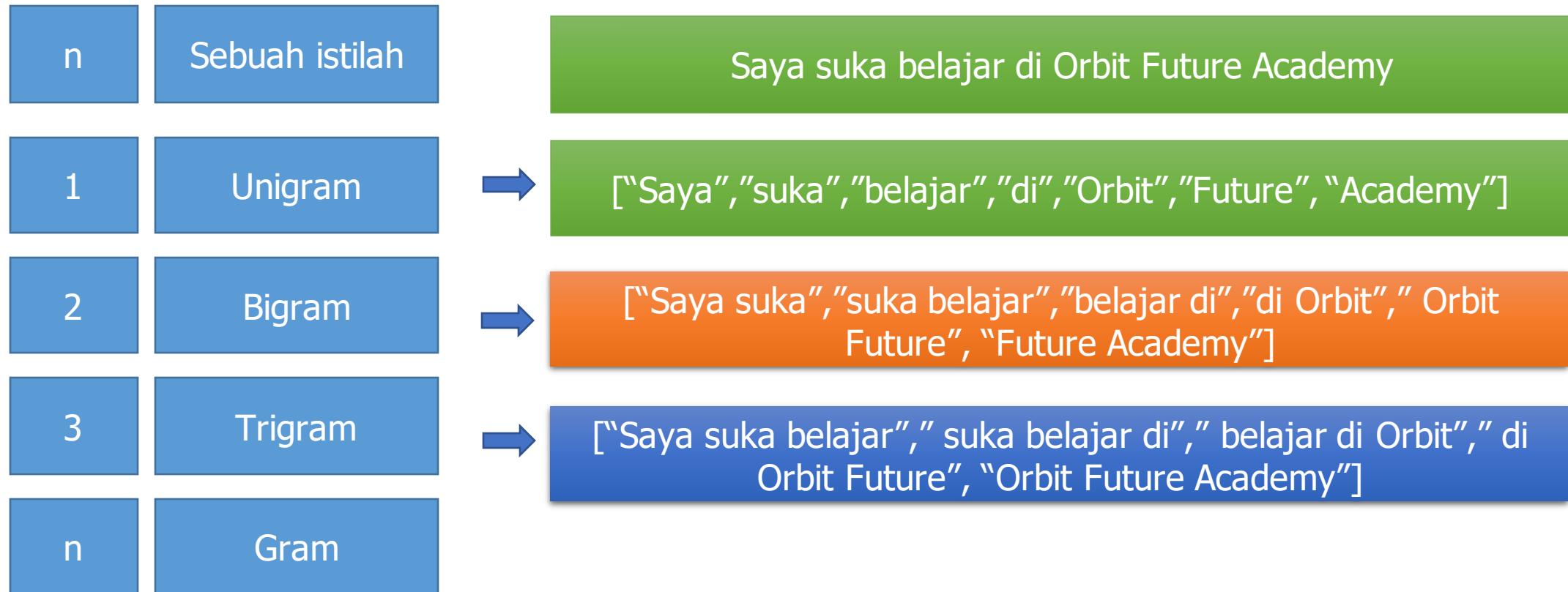
N-gram adalah rangkaian kata atau token yang berkesinambungan dalam sebuah dokumen yang digunakan untuk proses pada machine learning yang biasanya dijadikan fitur/atribut.

n

Sebuah variabel yang dapat memiliki nilai bilangan bulat positif termasuk 1,2,3 dan seterusnya. 'n' pada dasarnya mengacu pada kelipatan.

3. Jenis N-Gram

N-Gram yang dihasilkan





02 FEATURE FILTERING

- Feature Filtering
- Stop words

Feature Filtering

Proses untuk menghapus kata-kata atau simbol yang tidak penting, seperti HTML tag, URL, tanda baca , dll. [2]

1. Menghapus string, seperti @username, website , picture , tanda baca , dan karakter.

@username

picture (pic.twitter.com);

Punctuation (?!,)

website (https:// or http:// or www),

Misal,

@coach_orbit setuju minyak goreng menjadi #murah dan tidak langkah https://orbitfutureacademy.sch.id

Setelah Filtering



setuju minyak goreng menjadi #murah dan tidak langkah

[2] Prastyo, Pulung Hendro, et al. "Tweets responding to the Indonesian Government's handling of COVID-19: Sentiment analysis using SVM with normalized poly kernel." *Journal of Information Systems Engineering and Business Intelligence* 6.2 (2020): 112-122.

Feature Filtering

2. Stop word removal

Proses mengapus kata-kata yang tidak penting atau yang paling sering muncul pada sebuah kalimat, seperti kata penghubung

Di Ke Dengan

Stop word berbentuk kamus kata yang berisi kata-kata yang tidak penting.

Ketika kata-kata yang ada di stop word muncul pada sebuah kalimat, maka kata pada kalimat tersebut akan dihapus

| No | Stop Word list |
|-------|----------------|
| 1 | ke |
| 2 | di |
| 3 | itu |
| ... | |
| 1.000 | dengan |



Saya menyukai pakaian itu



Ini dihapus



03

FEATURE SELECTION

- Pengertian Seleksi Fitur
- Jenis-jenis Seleksi Fitur
- Penerapan Seleksi Fitur

Feature Selection (Seleksi Fitur)

Proses memilih kata-kata/fitur yang penting untuk diikutkan pada proses machine learning. Terdapat 3 pendekatan yang biasa digunakan untuk proses seleksi fitur, yaitu: [3]

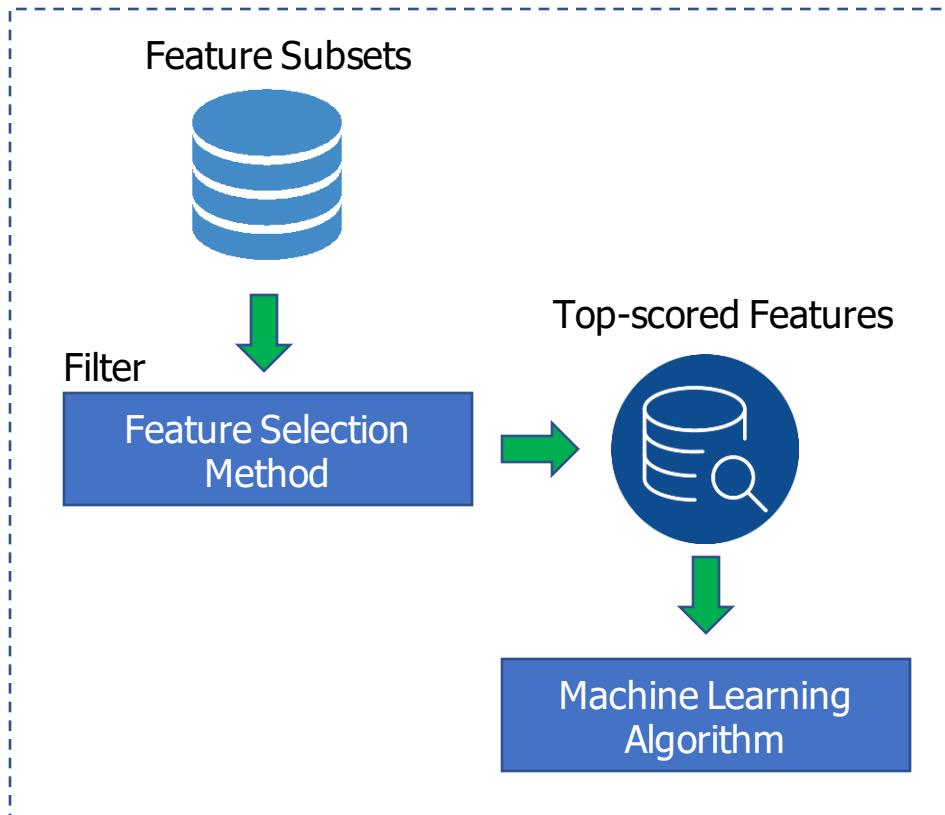
1. Metode Filter

2. Metode Wrapper

3. Metode Hybrid

[3] P. H. Prastyo, I. Ardiyanto and R. Hidayat, "A Review of Feature Selection Techniques in Sentiment Analysis Using Filter, Wrapper, or Hybrid Methods," *2020 6th International Conference on Science and Technology (ICST)*, 2020, pp. 1-6, doi: 10.1109/ICST50505.2020.9732885.

Feature Selection: Filter Method



Metode filter memberikan skor untuk setiap fitur, lalu fitur dengan skor tertinggi dipilih dan diberi peringkat. Metode filter bekerja secara terpisah dari algoritma machine learning.

Ada beberapa metode filter yang sudah diusulkan oleh para peneliti, seperti [Chi-square](#), [Information Gain](#), and [Query Expansion Ranking](#), etc.

Skema seleksi fitur berbasis filter

Feature Selection: Filter Method

Simulasi Seleksi Fitur Berbasis Filter

| No | Fitur | Skor |
|-----|--------|------|
| 1 | setuju | 0.8 |
| 2 | naik | 0.7 |
| 3 | Iuran | 0.5 |
| 4 | BPJS | 0.4 |
| 5 | saya | 0.11 |
| ... | | |
| 10 | besok | 0.10 |



| Saya | Setuju | Iuran | BPJS | Naik | .. | Besok |
|------|--------|-------|------|------|----|-------|
| 0.23 | 0.34 | 0.12 | 0.10 | 2.3 | .. | 0.2 |

Fitur
Vektor

Misal, dari 10 fitur yang ada, kita hanya mau mengambil 5 fitur, sehingga hanya fitur

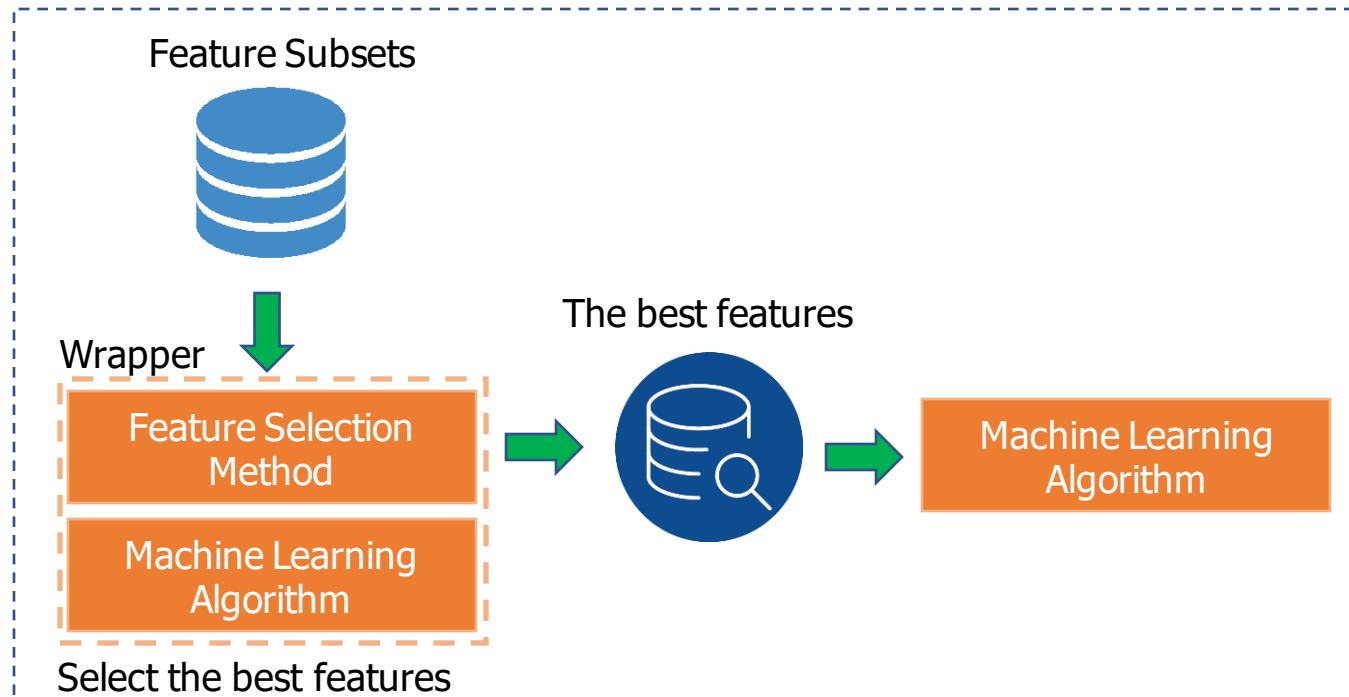
| Saya | Setuju | Iuran | BPJS | Naik |
|------|--------|-------|------|------|
| 0.23 | 0.34 | 0.12 | 0.10 | 2.3 |



Yang akan diikutkan pada proses machine learning karena memiliki skor yang tinggi.

Feature Selection: Wrapper Method

Metode **wrapper** adalah metode yang memilih fitur penting menggunakan algoritma optimasi dengan algoritma machine learning. Metode wrapper berinteraksi langsung dengan *classifier* dan menggunakan fungsi *fitness* untuk mengevaluasi fitur yang dipilih berdasarkan akurasi klasifikasi.



Ada beberapa metode wrapper yang pernah diusulkan oleh para peneliti, seperti **Genetic Algorithm**, **Particle Swarm Optimization**, and **Iterated Greedy Metaheuristic**, etc.

Skema seleksi fitur berbasis wrapper

Feature Selection: Wrapper Method

Simulasi Seleksi Fitur Berbasis Wrapper

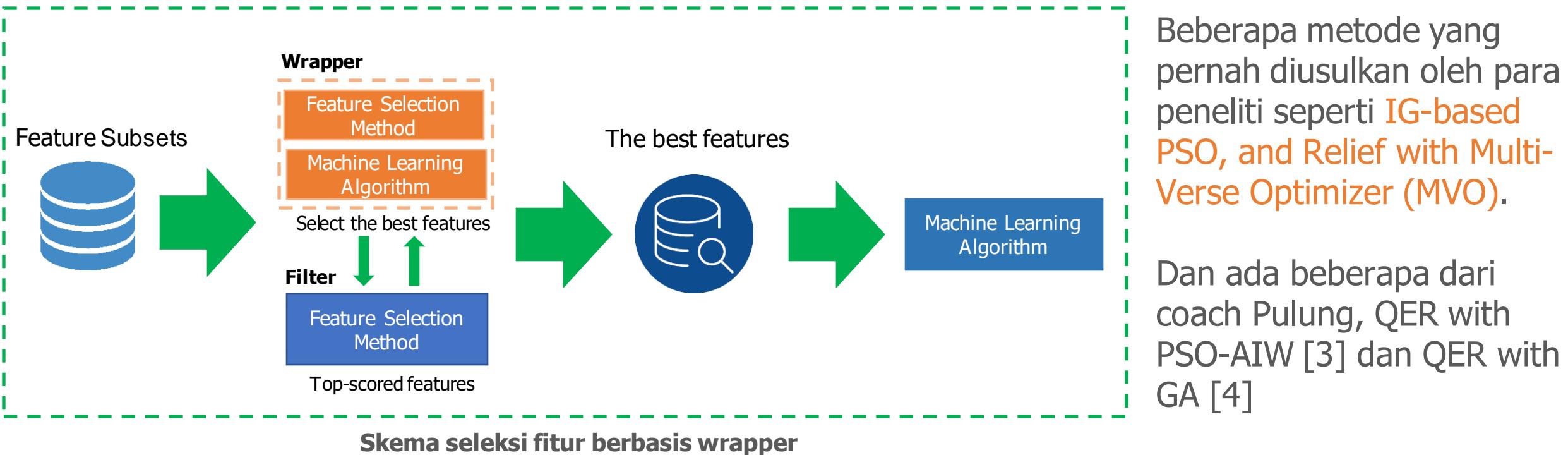
| Kombinasi Fitur | | | | | | | | Fitness |
|-----------------|--------|---------|--------|------|-------|----|-------|---------|
| | setuju | mereka | aparat | BPJS | turun | .. | tolak | |
| 1. | 1.2 | 0.34 | 0.12 | 0.10 | 2.3 | .. | 2.2 | 91.7 |
| | saya | setuju | iuran | BPJS | naik | .. | besok | 94.5 |
| 2. | 0.23 | 1.2 | 0.12 | 0.10 | 2.3 | .. | 0.2 | 95.7 |
| | setuju | sepakat | ikut | aksi | tolak | .. | baik | 95.7 |
| 3. | 1.2 | 0.90 | 0.12 | 0.10 | 3.1 | .. | 2.1 | 95.7 |

***Fitness** didapatkan dari nilai performa ML yang dijalankan oleh algoritme wrapper.
Perlu diingat bahwa menentukan fungsi fitness tergantung kasus yang ingin kita selesaikan.

Kombinasi fitur ke-3 merupakan kombinasi fitur terbaik sehingga,
fitur-fitur tersebut digunakan pada tahap Machine Learning

Feature Selection: Hybrid Method

Metode hybrid adalah **kombinasi dari metode filter dan wrapper**. Pertama, metode filter dijalankan untuk mengurangi fitur dengan memilih fitur-fitur yang memiliki nilai tertinggi. Kemudian, metode wrapper digunakan untuk mencari fitur terbaik dari hasil seleksi metode filter.



[3] Prastyo, Pulung Hendro, Risanuri Hidayat, and Igi Ardiyanto. "Enhancing sentiment classification performance using hybrid Query Expansion Ranking and Binary Particle Swarm Optimization with Adaptive Inertia Weights." *ICT Express* (2021).

[4] Prastyo, Pulung Hendro, Igi Ardiyanto, and Risanuri Hidayat. "A Combination of Query Expansion Ranking and GA-SVM for Improving Indonesian Sentiment Classification Performance." *Procedia Computer Science* 189 (2021): 108-115.

Feature Selection: Hybrid Method

Simulasi Seleksi Fitur Berbasis Hybrid

| saya | setuju | iuran | BPJS | naik | .. | nesok |
|------|--------|-------|------|------|----|-------|
| 0.23 | 0.34 | 0.12 | 0.10 | 2.3 | .. | 0.2 |

Tanpa seleksi fitur

↓

| saya | setuju | iuran | BPJS | naik | .. | besok |
|------|--------|-------|------|------|----|-------|
| 0.23 | 0.34 | 0.12 | 0.10 | 2.3 | .. | 0.2 |

Seleksi fitur berbasis Filter.
Memilih 5 fitur dengan skor tertinggi

↓

| saya | setuju | iuran | BPJS | naik | .. | besok |
|------|--------|-------|------|------|----|-------|
| 0.23 | 0.34 | 0.12 | 0.10 | 2.3 | .. | 0.2 |

Seleksi fitur berbasis Wrapper.
Memilih fitur terbaik dari hasil
seleksi berbasis filter

██████████ Dihapus/terseleksi

Feature Selection

Tabel perbandingan metode fitur seleksi [3]

| Metode | Kelebihan | Kekurangan |
|---------|--|--|
| Filter | Sederhana, independen dari metode klasifikasi yang digunakan, dan lebih cepat dari teknik lainnya. | Masih terdapat fitur yang tidak sesuai, dan fitur yang berlebihan. |
| Wrapper | Akurasi tinggi, dapat meningkatkan kinerja pengklasifikasi secara signifikan. | Lambat, kompleks. |
| Hybrid | Hasil yang sangat baik, mencapai kinerja yang signifikan | Lebih kompleks, Biaya waktu komputasi yang tinggi, membutuhkan GPU untuk mendapatkan performa terbaik. |

[5] P. H. Prastyo, I. Ardiyanto and R. Hidayat, "A Review of Feature Selection Techniques in Sentiment Analysis Using Filter, Wrapper, or Hybrid Methods," *2020 6th International Conference on Science and Technology (ICST)*, 2020, pp. 1-6, doi: 10.1109/ICST50505.2020.9732885.



04 WORD CLOUD

- Pengertian Word Cloud
- Penggunaan Word Cloud

Word Cloud

Gambar yang menunjukkan daftar kata-kata yang digunakan dalam sebuah teks, umumnya semakin banyak kata yang digunakan semakin besar ukuran kata tersebut dalam gambar.



Penggunaan Word Cloud

Visualisasi kata yang paling sering digunakan atau diperbincangkan pada suatu topik tertentu. Misal, pada domain **sentiment analysis** kita ingin melihat kata yang paling sering muncul ketika masyarakat memberikan opini positif ataupun negatif pada **kasus omnibus law**.

POSITIVE

```
1 # Create and generate a word cloud image:  
2 wordcloud = WordCloud().generate(str(data_positive.clean_text))  
3  
4 # Display the generated image:  
5 plt.imshow(wordcloud, interpolation='bilinear')  
6 plt.axis("off")  
7 plt.show()
```



NEGATIVE

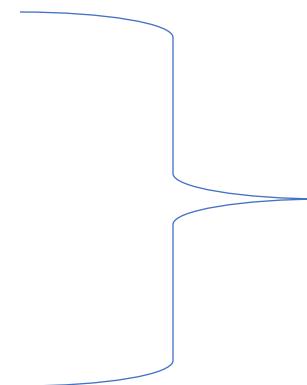
```
: 1 # Create and generate a word cloud image:  
2 wordcloud = WordCloud().generate(str(data_negative.clean_text))  
3  
4 # Display the generated image:  
5 plt.imshow(wordcloud, interpolation='bilinear')  
6 plt.axis("off")  
7 plt.show()
```



Word Cloud Tools



Word Cloud



```
1 from wordcloud import WordCloud  
2 import matplotlib.pyplot as plt  
3 |
```

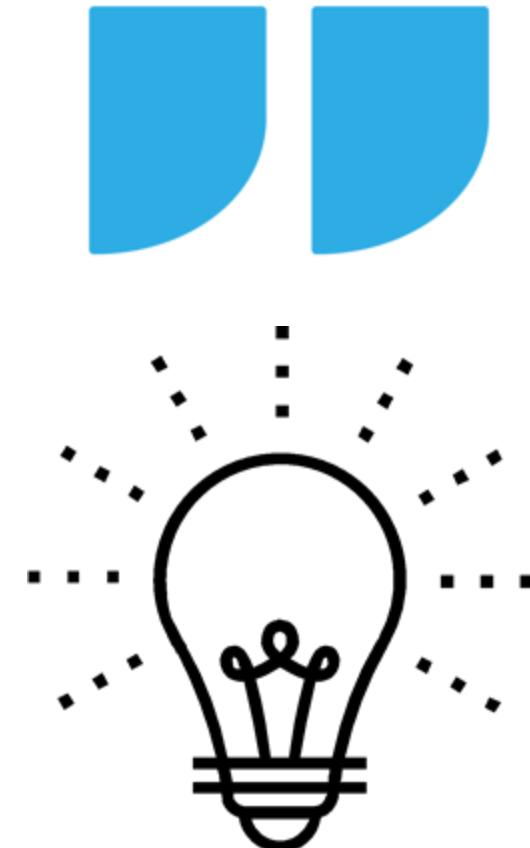


05 KESIMPULAN

- Ringkasan
- Kuis

Ringkasan

1. **Feature Extraction** adalah proses mengubah kata menjadi bentuk vektor agar bisa diproses oleh algoritme *machine learning*.
2. Ada dua jenis contoh **Feature Extraction** yang bisa digunakan,yaitu BoW dan TF-IDF.
3. **N-gram** adalah rangkaian kata atau token yang berkesinambungan dalam sebuah dokumen yang digunakan untuk proses pada machine learning yang biasanya dijadikan fitur/atribut.
4. **Feature Filtering** adalah proses untuk menghapus kata-kata atau simbol yang tidak penting, seperti html tag, url, tanda baca.
5. **Feature Selection** adalah proses memilih kata-kata/fitur yang penting untuk diikutkan pada proses machine learning.
6. **Word Cloud** adalah gambar yang menunjukkan daftar kata-kata yang digunakan dalam sebuah teks



Kuis

Pertanyaan

Proses mengubah teks menjadi vektor agar dapat diproses oleh algoritme Machine Learning ?

- A. Feature Selection
- B. Feature Filtering
- C. Feature Extraction
- D. Stopword Removal



Kuis

Pertanyaan

Proses mengubah teks menjadi vektor agar dapat diproses oleh algoritme Machine Learning ?

- A. Feature Selection
- B. Feature Filtering
- C. Feature Extraction**
- D. Stopword Removal



Jawaban: C



TERIMA KASIH

Orbit Future Academy

PT Orbit Ventura Indonesia
Center of Excellence (Jakarta Selatan)
Gedung Veteran RI, Lt.15
Unit Z15-002, Plaza Semanggi
Jl. Jenderal Sudirman Kav.50, Jakarta
12930, Indonesia

- Jakarta Selatan/Pusat
- Jakarta Barat/BSD
- Kota Bandung
- Kab. Bandung
- Jawa Barat

Hubungi Kami

Director of Sales & Partnership
ira@orbitventura.com
+62 858-9187-7388

Social Media

-  [Orbit Future Academy](#)
-  [@OrbitFutureAcademyIn1](#)
-  [OrbitFutureAcademy](#)
-  [Orbit Future Academy](#)



AI Mastery Course



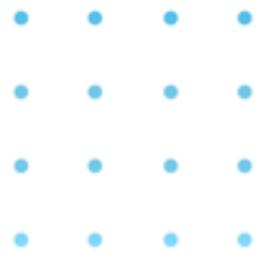
Module 10

Natural Language Processing (NLP)

Section

Text Classification





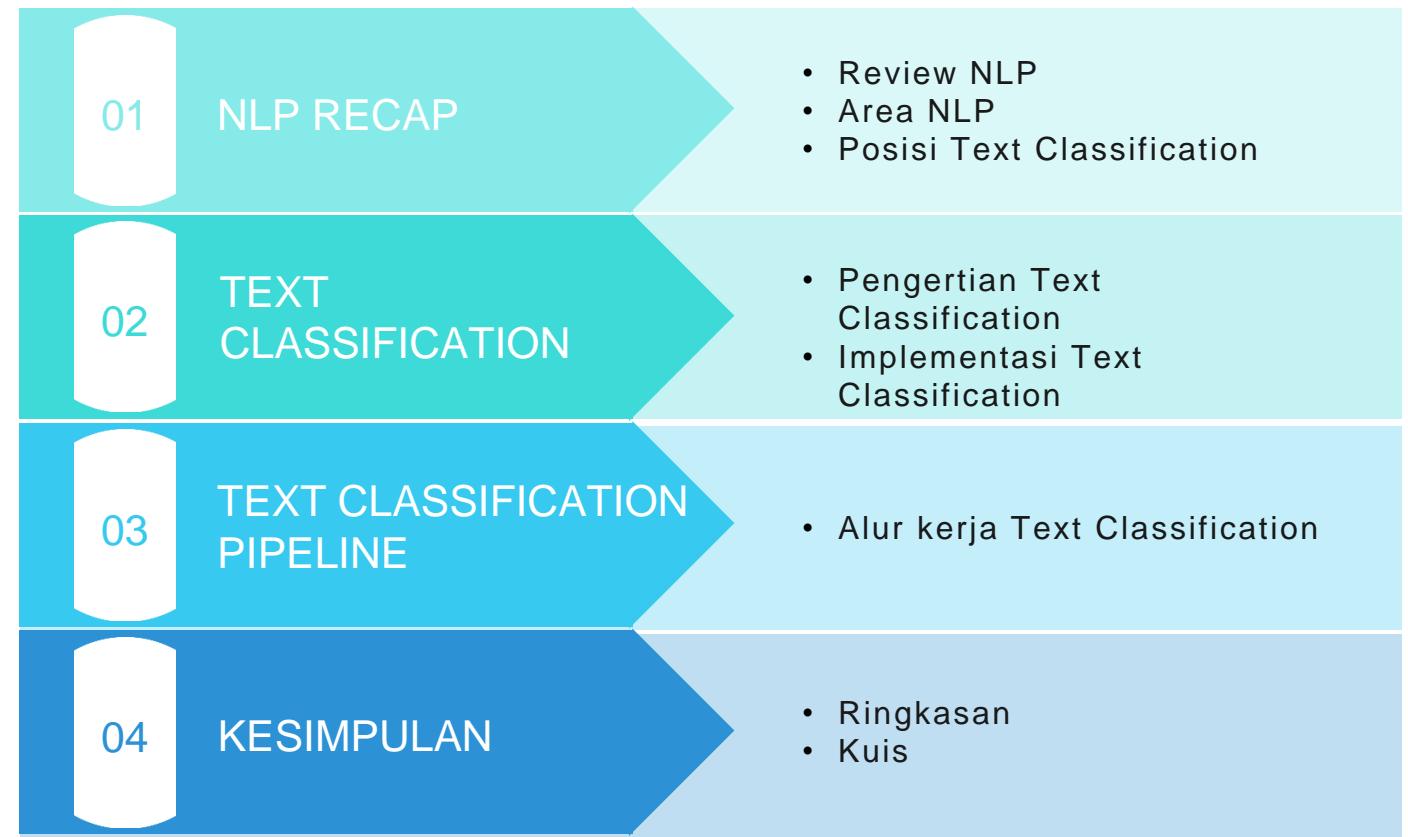
Learning Objectives

Di akhir modul ini, Anda akan mendapatkan:

- Mengetahui pengertian Text Classification
- Mengetahui area Text Classification
- Memahami konsep dasar proses text classification (Text Classification Pipeline)
- Mampu mengimplementasikan Text Classification dengan baik menggunakan Bahasa Pemrograman Python.



Agenda

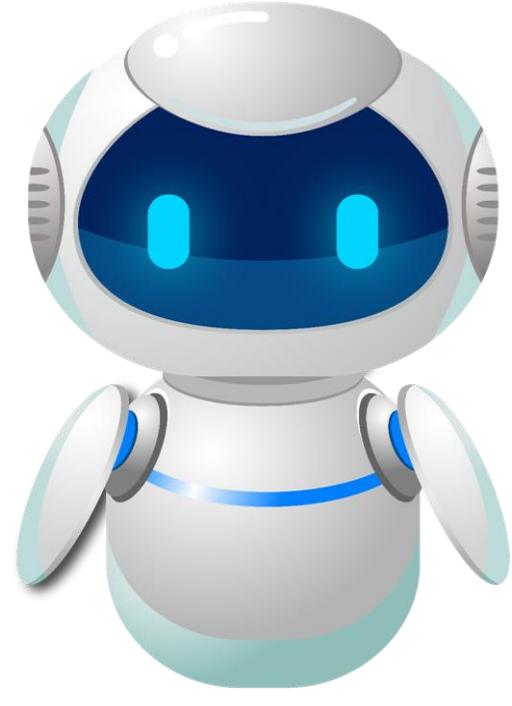




01 NLP RECAP

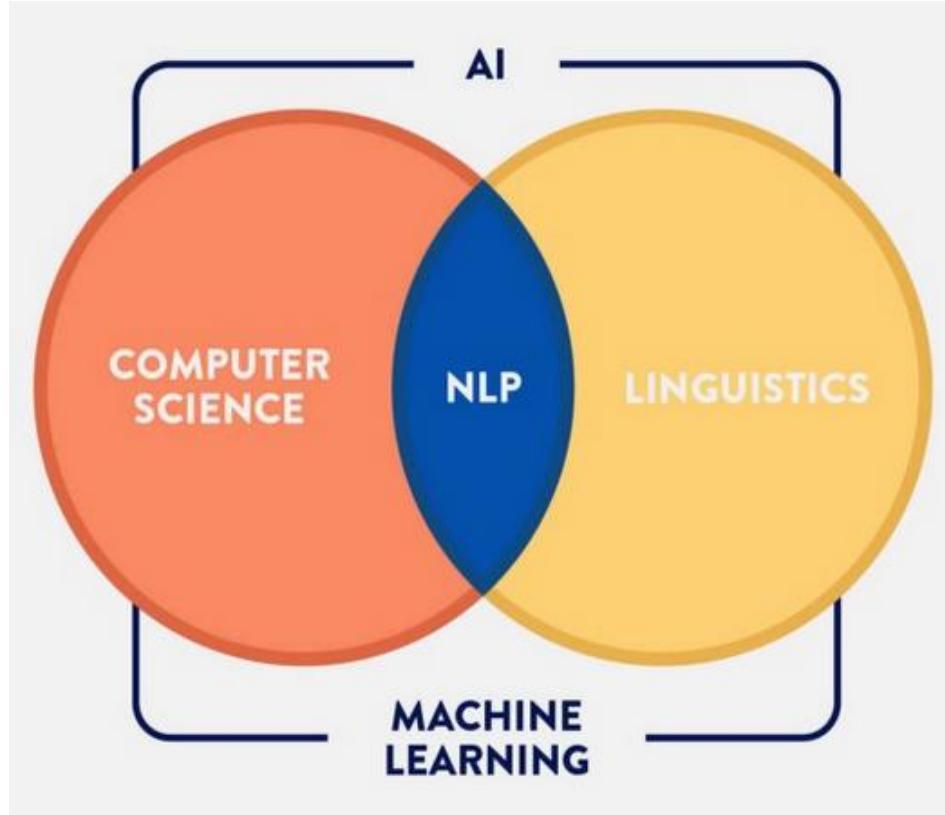
- Review NLP
- Area NLP
- Posisi Text Classification

NLP Recap



NLP (Natural Language Processing)

NLP Recap

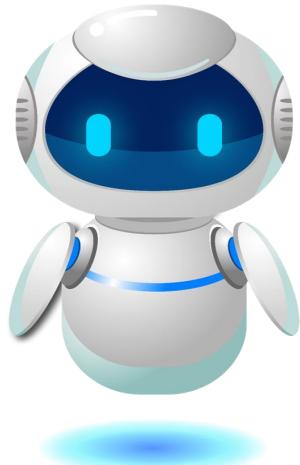


Domain AI yang berhubungan dengan interaksi antara mesin dan manusia menggunakan **bahasa alami**. NLP menggunakan data tidak terstruktur: textual maupun suara.

Data teks dan bahasa dapat memiliki isi dan ukuran yang sangat beragam sehingga lebih menantang dibandingkan data tabular biasa.

NLP Area Recap

Let's go!



Question Answering Systems (QAS)

Summarization

Machine Translation

Speech Recognition

Text classification



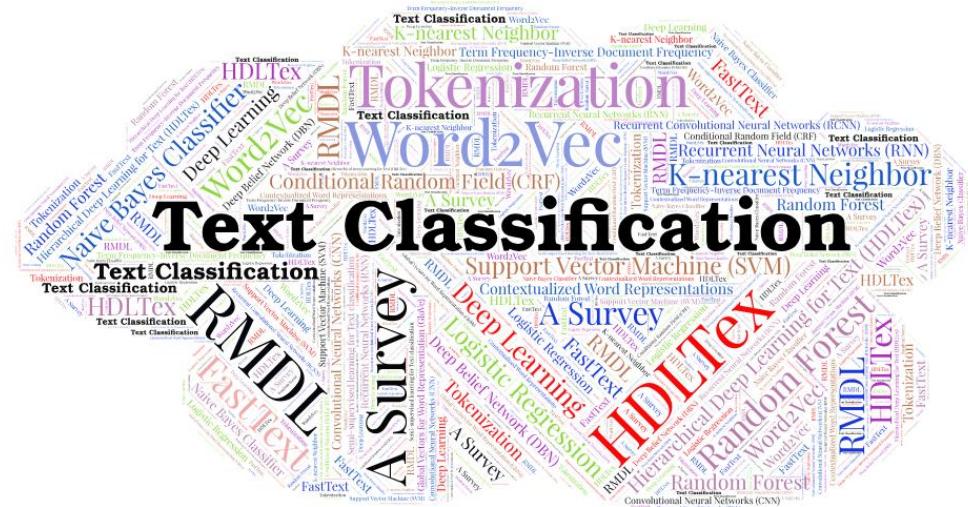
Kita akan
belajar area ini



02 TEXT CLASSIFICATION

- Pengertian Text Classification
- Implementasi Text Classification
- Pendekatan Text Classification

Text Classification

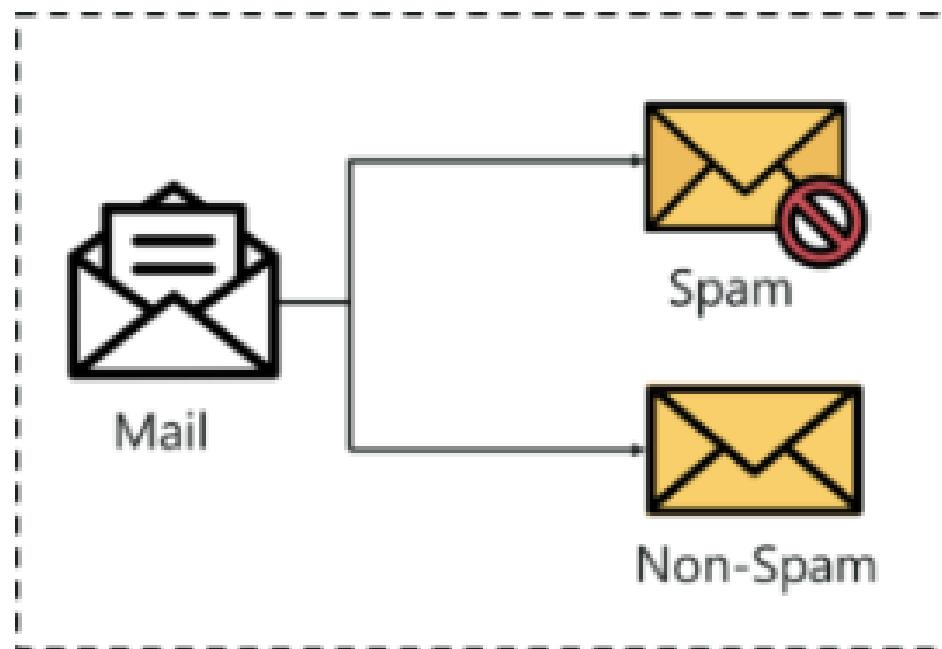


Text classification / klasifikasi teks adalah proses pemberian tag atau kategori ke teks menurut isinya. Klasifikasi teks dapat digunakan untuk mengatur, menyusun, dan mengkategorikan hampir semua hal.

Misalnya, artikel baru dapat diatur berdasarkan topik, percakapan obrolan dapat diatur berdasarkan bahasa, penyebutan merek dapat diatur berdasarkan sentimen, dan sebagainya.

Implementasi Text Classification

Email/SMS Classification



Proses menentukan apakah email/sms tersebut spam atau non-spam.

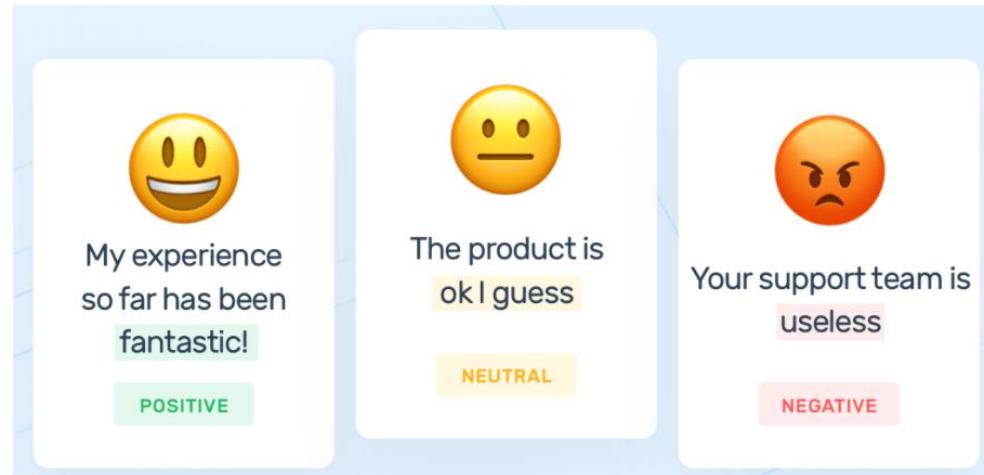
Baca selanjutnya:

E. G. Dada, J. S. Bassi, H. Chiroma, S. M. Abdulhamid, A. O. Adetunmbi, and O. E. Ajibuwu, "Machine learning for email spam filtering: review, approaches and open research problems," *Heliyon*, vol. 5, no. 6, p. e01802, Jun. 2019.

<https://doi.org/10.1016/j.heliyon.2019.e01802>

Implementasi Text Classification

Sentiment Analysis



Menentukan polaritas sebuah teks, apakah bernilai positif atau negatif.

- Product review
- Movie review
- Kebijakan pemerintah, dll

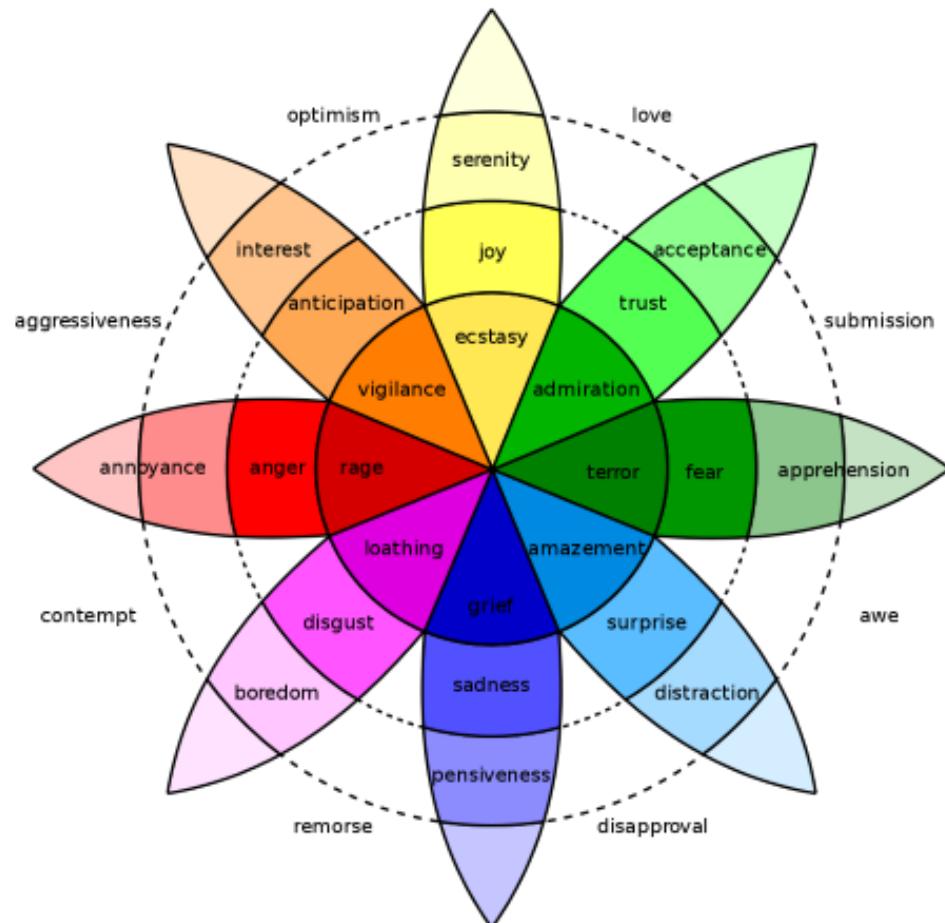
Baca selanjutnya:

Z. Drus and H. Khalid, “Sentiment Analysis in Social Media and Its Application: Systematic Literature Review,” Procedia Computer Science, vol. 161, pp. 707–714, 2019.

<https://doi.org/10.1016/j.procs.2019.11.174>

Implementasi Text Classification

Emotion Detection



Bagian sentiment analysis. Menentukan emosi dari teks. Apakah sebuah teks berindikasi, senang, sedih, marah, dll.

Baca selanjutnya:

A. Chowanda, R. Sutoyo, Meiliana, and S. Tanachutiwat, "Exploring Text-based Emotions Recognition Machine Learning Techniques on Social Media Conversation," Procedia Computer Science, vol. 179, pp. 821–828, 2021.

<https://doi.org/10.1016/j.procs.2021.01.099>

Pendekatan Text Classification



Machine Learning



Lexicon-based

P. H. Prastyo, I. Ardiyanto and R. Hidayat, "A Review of Feature Selection Techniques in Sentiment Analysis Using Filter, Wrapper, or Hybrid Methods," *2020 6th International Conference on Science and Technology (ICST)*, 2020, pp. 1-6,
doi: 10.1109/ICST50505.2020.9732885.

Machine Learning Approach



Proses klasifikasi teks dengan melatih mesin untuk mempelajari pola pada kumpulan teks.

Kelebihan:

- Mampu menentukan polaritas sebuah teks / sentiment baru (unseen data)Learning

Kekurangan:

- Dimensi fitur yang tinggi
- Fitur noise
- Pelabelan data

| Text | Sentiment |
|---|-----------|
| Saya suka dan setuju kebijakan pemerintah | Positif |
| Saya menolak kebijakan pemerintah | Negatif |

Lexicon-Based Approach



Proses klasifikasi teks dengan menggunakan kamus kata yang sudah didefinisikan sebelumnya.

Kelebihan:

- Tanpa pelabelan data
- Proses komputasi lebih cepat

Kekurangan:

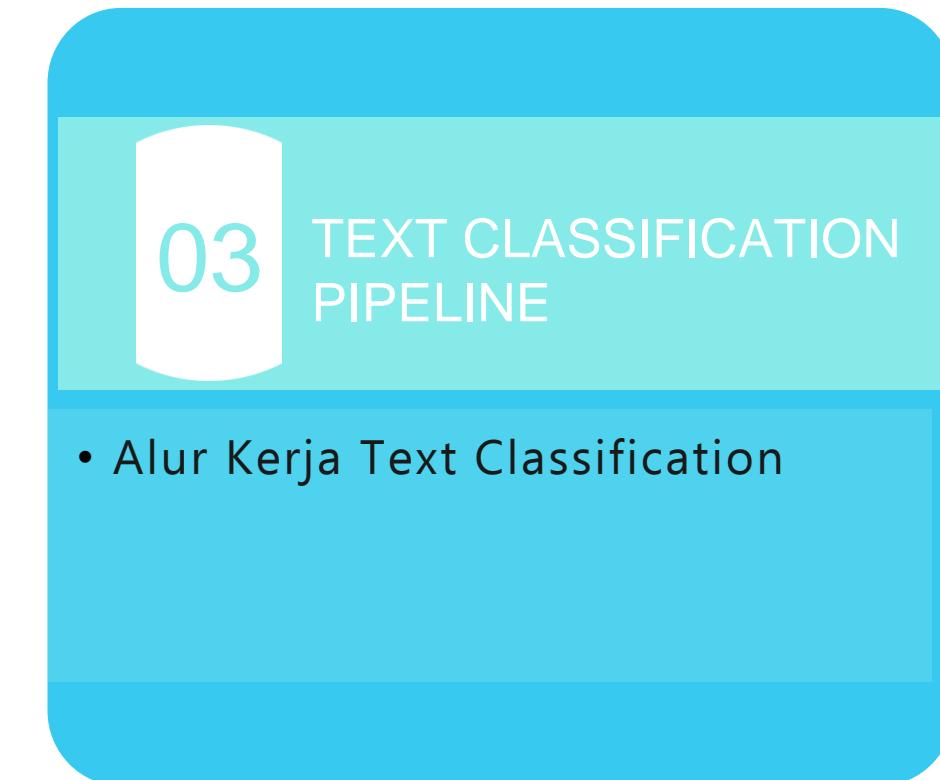
- Harus mendaftarkan semua kata dalam kamus kata. Jika belum terdaftar, sangat mempengaruhi performa klasifikasi.

Saya suka dan setuju kebijakan pemerintah

$$+1 \quad +5 \quad +5 \quad +1 \quad +1 = 13 \text{ (Positif)}$$

Saya menolak kebijakan pemerintah

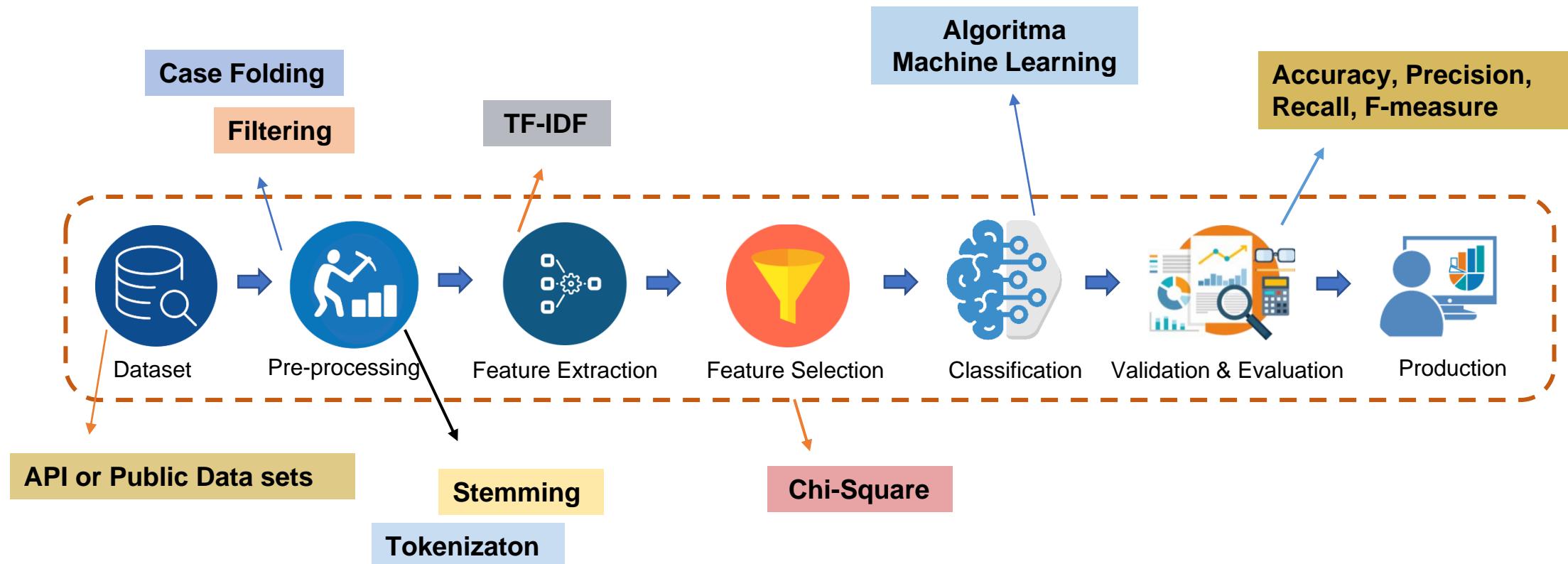
$$+1 \quad -5 \quad +1 \quad +1 = -2 \text{ (Negatif)}$$



03 TEXT CLASSIFICATION PIPELINE

- Alur Kerja Text Classification

Proses Text Classification



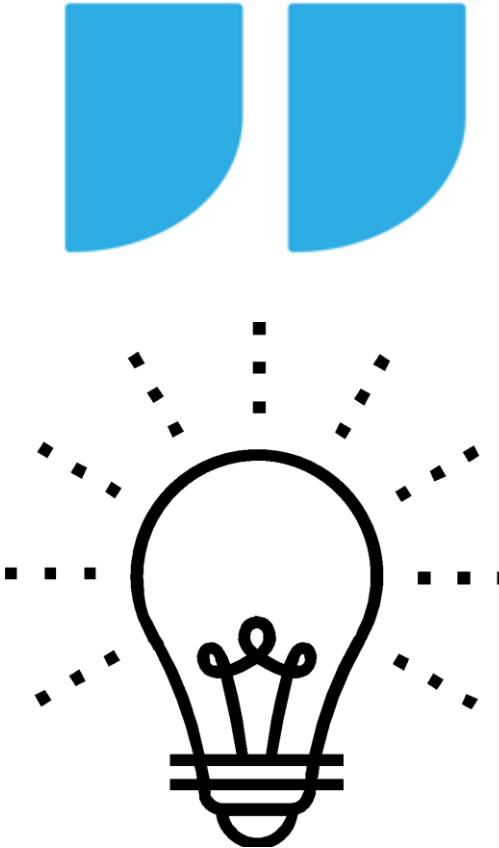


05 KESIMPULAN

- Ringkasan
- Kuis

Ringkasan

1. Text Classification/ Klasifikasi teks adalah proses pemberian tag atau kategori ke teks menurut isinya. Klasifikasi teks dapat digunakan untuk mengatur, menyusun, dan mengkategorikan hampir semua hal.
2. Macam-macam area implementasi Text Classification, seperti Deteksi Spam, analisis sentimen, deteksi/klasifikasi emosi
3. Dua pendekatan umum Teks Classification, yaitu Machine Learning dan Lexicon



Kuis

Pertanyaan

Salah satu implementasi dari Text Classification adalah ?

- A. Object Detection
- B. Image Detection
- C. Emotion Detection
- D. Face Detection



Kuis

Pertanyaan

Salah satu implementasi dari Text Classification adalah ?

- A. Object Detection
- B. Image Detection
- C. Emotion Detection**
- D. Face Detection



Jawaban: C



TERIMA KASIH

Orbit Future Academy

PT Orbit Ventura Indonesia
Center of Excellence (Jakarta Selatan)
Gedung Veteran RI, Lt.15
Unit Z15-002, Plaza Semanggi
Jl. Jenderal Sudirman Kav.50, Jakarta
12930, Indonesia

- Jakarta Selatan/Pusat
- Jakarta Barat/BSD
- Kota Bandung
- Kab. Bandung
- Jawa Barat

Hubungi Kami

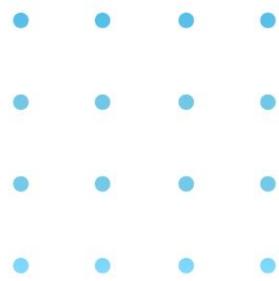
Director of Sales & Partnership
ira@orbitventura.com
+62 858-9187-7388

Social Media

- Orbit Future Academy
- @OrbitFutureAcademyIn1
- OrbitFutureAcademy
- Orbit Future Academy



AI Mastery Course

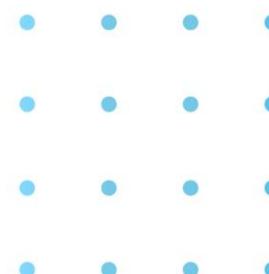


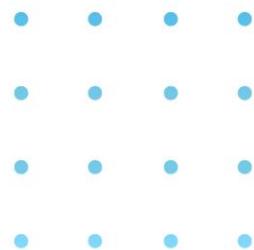
Module 10

Natural Language Processing (NLP)

Section

Word Embedding and
Deep Learning for NLP

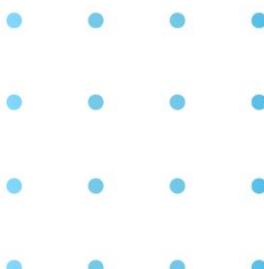




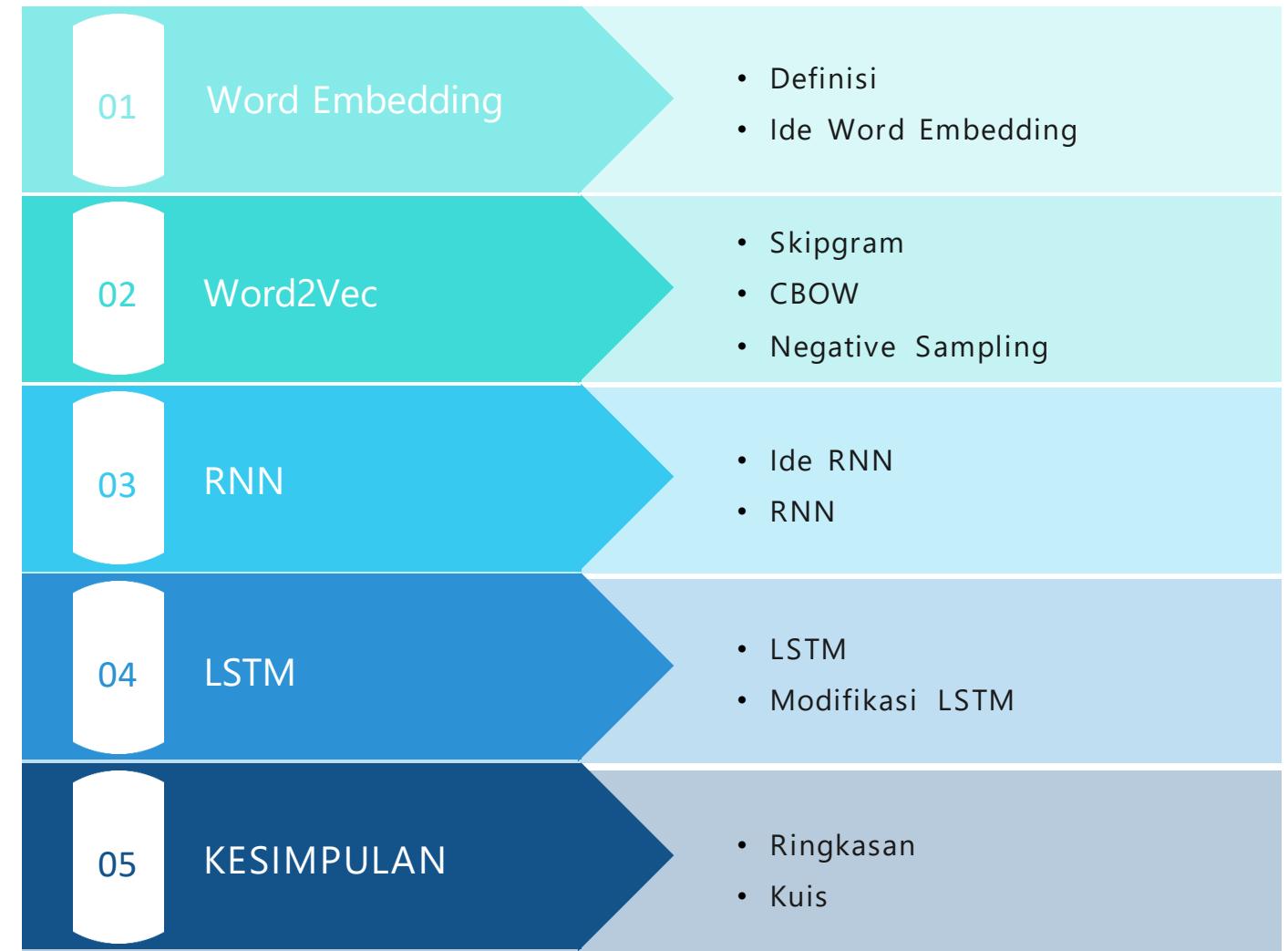
Learning Objectives

Di akhir modul ini, Anda akan mendapatkan:

- Dapat memahami ide dan alasan dibutuhkannya word embedding
- Dapat memahami intuisi word2vec dengan metode skipgram dan CBOW
- Dapat memahami negative sampling pada Word2Vec
- Dapat memahami Recurrent Neural Network dan batasannya
- Dapat memahami Long short-term memory (LSTM)
- Dapat mengimplementasikan setiap metode dengan python



Agenda





01

Word Embedding

- Definisi
- Ide Word Embedding

Machine Learning Berjumpa dengan Data berupa Bahasa (teks)



Text
Data

Tabular
Data

Recall: Bag of words dan TF-IDF

Kekurangan:

1. Sparse, setiap kata pada dictionary (kamus) merupakan fitur. Hal ini bertambah apabila menggunakan bigram, trigram dst.

| | aa | aamiiin | aamiin | ab | abadi |
|---|-----|---------|--------|-----|-------|
| 0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 1 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 2 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 3 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 4 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |

Banyak kolom bernilai nol,
sehingga data menjadi sparse

2. Low Semantics, tidak memiliki kemampuan lebih untuk menangkap makna (semantics).

Selain itu pada ekstraksi fitur, baik BoW dan TF-IDF tidak memperhatikan urutan

contoh: "Saya mengantuk, tadi tidak tidur"

"Saya tidak mengantuk, tadi tidur"

Pada BoW dan TF-IDF, representasi mereka sama

| | saya | mengantuk | tadi | tidak | tidur |
|---|------|-----------|------|-------|-------|
| Saya mengantuk, tadi tidak tidur | 1 | 1 | 1 | 1 | 1 |
| Saya tidak mengantuk, tadi tidur | 1 | 1 | 1 | 1 | 1 |

Recall: Bag of words dan TF-IDF

3. Out of Vocabulary Problem, Jika tidak terdapat pada kamus, kata baru tidak memiliki fitur

Sebelumnya kita hanya memiliki fitur berikut: 'saya', 'mengantuk', 'tadi', 'tidak', 'tidur'

Ketika muncul kalimat baru: "Saya mengantuk, tadi tidak tidur siang"

| | saya | mengantuk | tadi | tidak | tidur |
|-----------|------|-----------|------|-------|-------|
| saya | 1 | 0 | 0 | 0 | 0 |
| mengantuk | 0 | 1 | 0 | 0 | 0 |
| tadi | 0 | 0 | 1 | 0 | 0 |
| tidak | 0 | 0 | 0 | 1 | 0 |
| tidur | 0 | 0 | 0 | 0 | 1 |
| siang | ? | ? | ? | ? | ? |

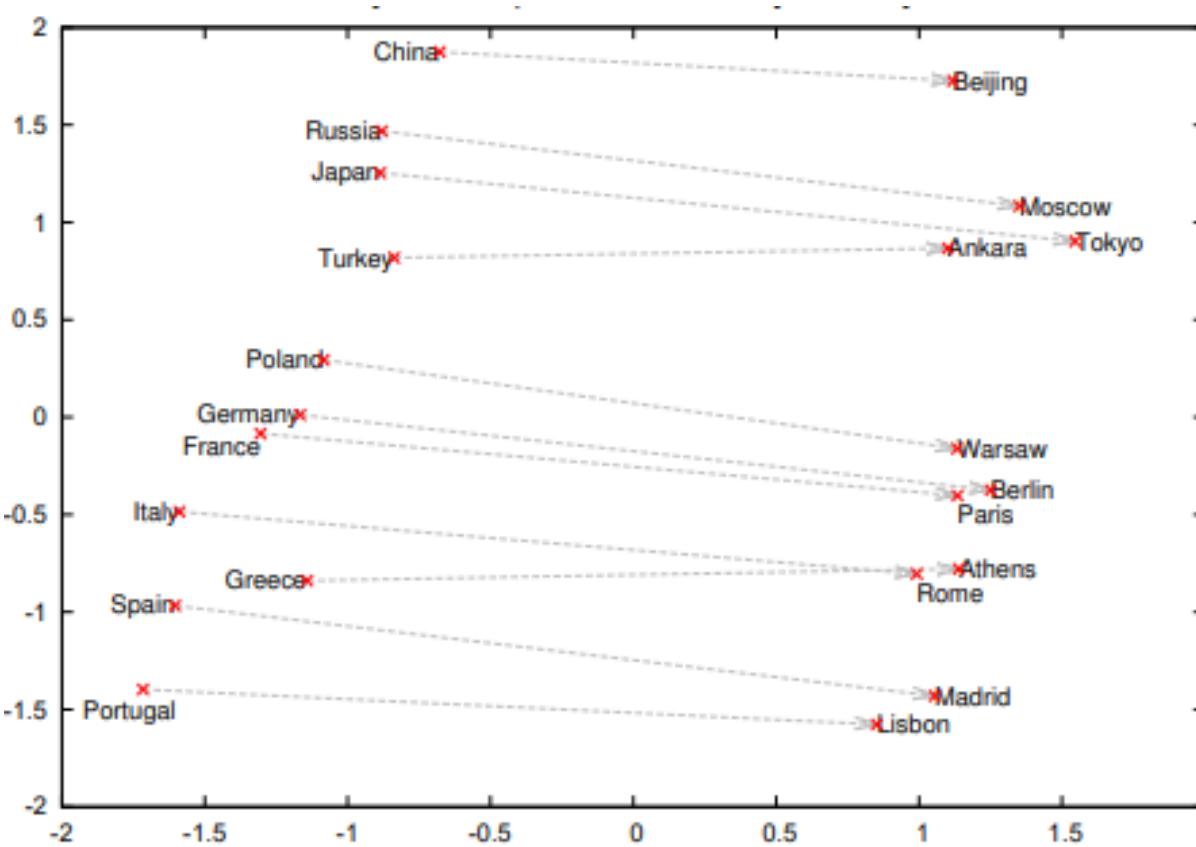
kata "siang" tidak terdapat pada kamus sebelumnya

Namun, menurutmu apa keuntungan dari BoW dan TF-IDF dan apa contoh penerapannya ?

Word Embedding

Word Embedding hadir guna menyelesaikan permasalahan sebelumnya
Setiap kata dapat direpresentasikan pada vector space dalam dimensi tertentu.

Kata-kata yang memiliki konteks yang mirip saling berdekatan



Karakteristik Word Embedding yang diinginkan:

1. Ukuran fitur yang lebih pendek
2. Dense (tidak memiliki banyak nol pada

| | abah | ... | future | orbit | ... | tidur |
|-------|------|-----|--------|-------|-----|-------|
| orbit | 0 | 0 | 0 | 1 | 0 | 0 |
| tidur | 0 | 0 | 0 | 0 | 0 | 1 |
| ... | | | | | | |

Contoh di atas adalah contoh fiturisasi dengan BoW, dan terdapat banyak nilai nol.
Pada Word Embedding, diharapkan datanya berupa:

Misalkan:

"tidur" = [0.12, 0.22, 0.11, 0.75,...]

Yakni setiap elemen memiliki nilai

Hal ini menghindari data 'sparse'

Word Embedding

Zero Padding & Truncation

Zero padding memberikan nilai nol pada kalimat yang kurang dari panjang kalimat yang telah diset

Sedangkan Truncation memotong kata pada suatu kalimat yang ukurannya lebih dari Panjang kalimat yang telah diset

| No | kalimat | Token kalimat | Jumlah kata | Post-Process |
|----|--|---|-------------|---|
| 1 | karena saya lagi di surabaya sampai besok hari | ['karena', 'saya', 'lagi', 'di', 'Surabaya', 'sampai', 'besok', 'hari'] | 8 | ['karena', 'saya', 'lagi', 'di', 'Surabaya', 'sampai', 'besok'] |
| 2 | udah lama ga bimbingan | ['udah', 'lama', 'ga', 'bimbingan'] | 4 | ['udah', 'lama', 'ga', 'bimbingan', 0, 0, 0] |

Setiap token kata dijadikan vocabulary dan diakses dengan **Indeks**

| Indeks | Kata | indeks | Kata | indeks | Kata |
|--------|-----------|--------|--------|--------|----------|
| 1 | besok | 5 | hari | 9 | sampai |
| 2 | bimbingan | 6 | karena | 10 | saya |
| 3 | di | 7 | lagi | 11 | sudah |
| 4 | ga | 8 | lama | 12 | surabaya |



02 Word2Vec

- Skipgram
- CBOW
- Negative Sampling



Word2Vec

Ide : Daripada menghitung kemunculan kata seperti Bag of words dan TF-IDF. Bagaimana memprediksi kemunculan kata-kata di sekitar suatu kata (kasus klasifikasi)

Bagaimana ini bekerja ?

- Word2Vec diperoleh dengan melatih shallow neural network
- Shallow merujuk pada istilah pemanfaatan satu hidden layer dari neural network
- Kita hanya mengambil weights dari classifier dan dijadikan word embedding
- model dari shallow neural network tidak digunakan
- Jumlah elemen vector atau disebut dimensi dari word embedding dapat ditentukan
- Banyaknya elemen vector atau dimensi diatur dengan menentukan jumlah hidden layer nantinya

Misalnya, dimensi ditentukan sebanyak 5 , maka untuk word embedding dari kata "tidur" ialah:

[0.12, 0.22, 011, 0,75,]

Yakni memiliki 5 elemen / dimensi

Terdapat dua cara paling umum untuk training data dengan Word2Vec:

1. Skip-gram
2. CBOW

Word2Vec - Skipgram

Window size = 2

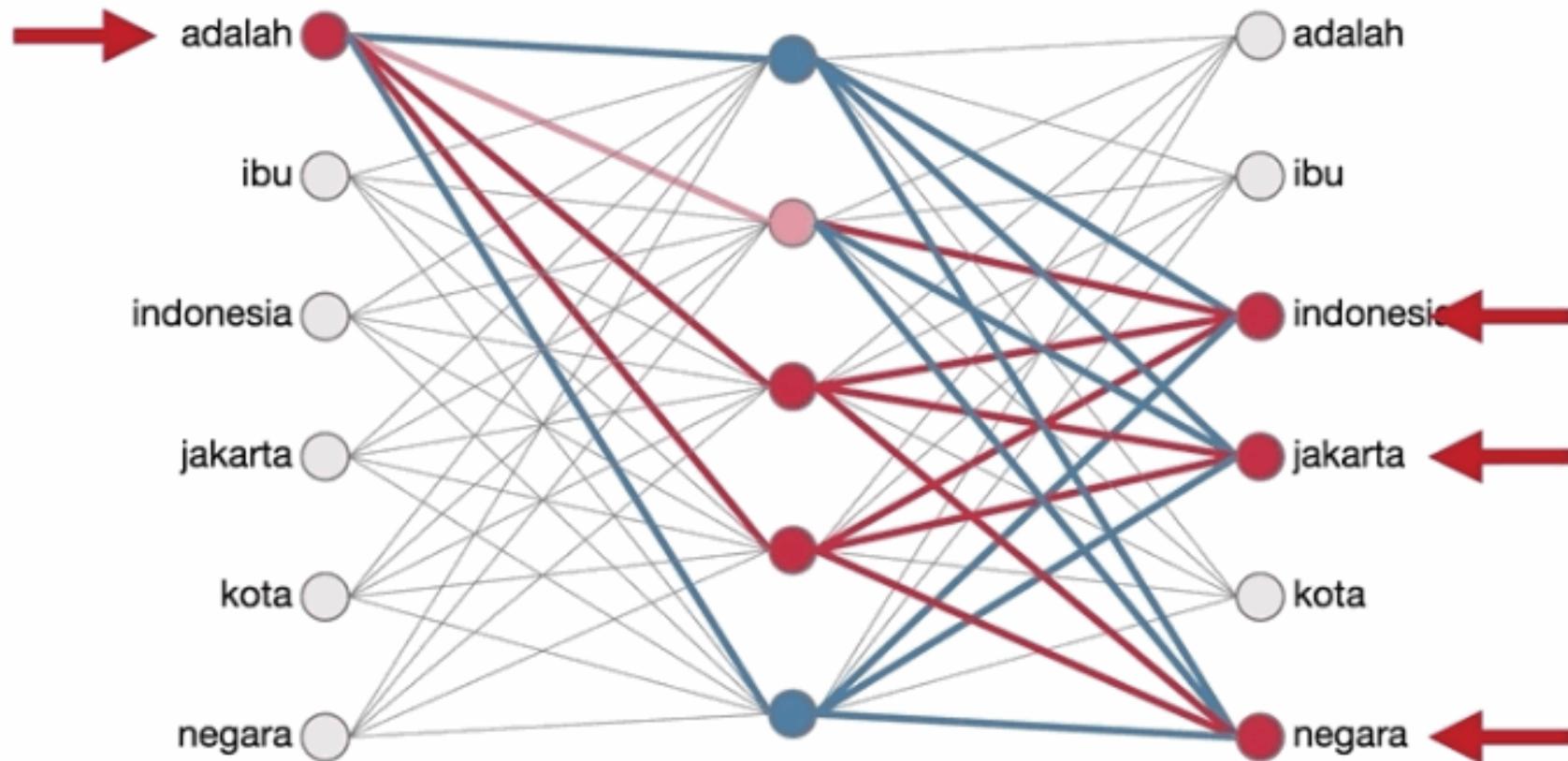


Word2Vec - Skipgram

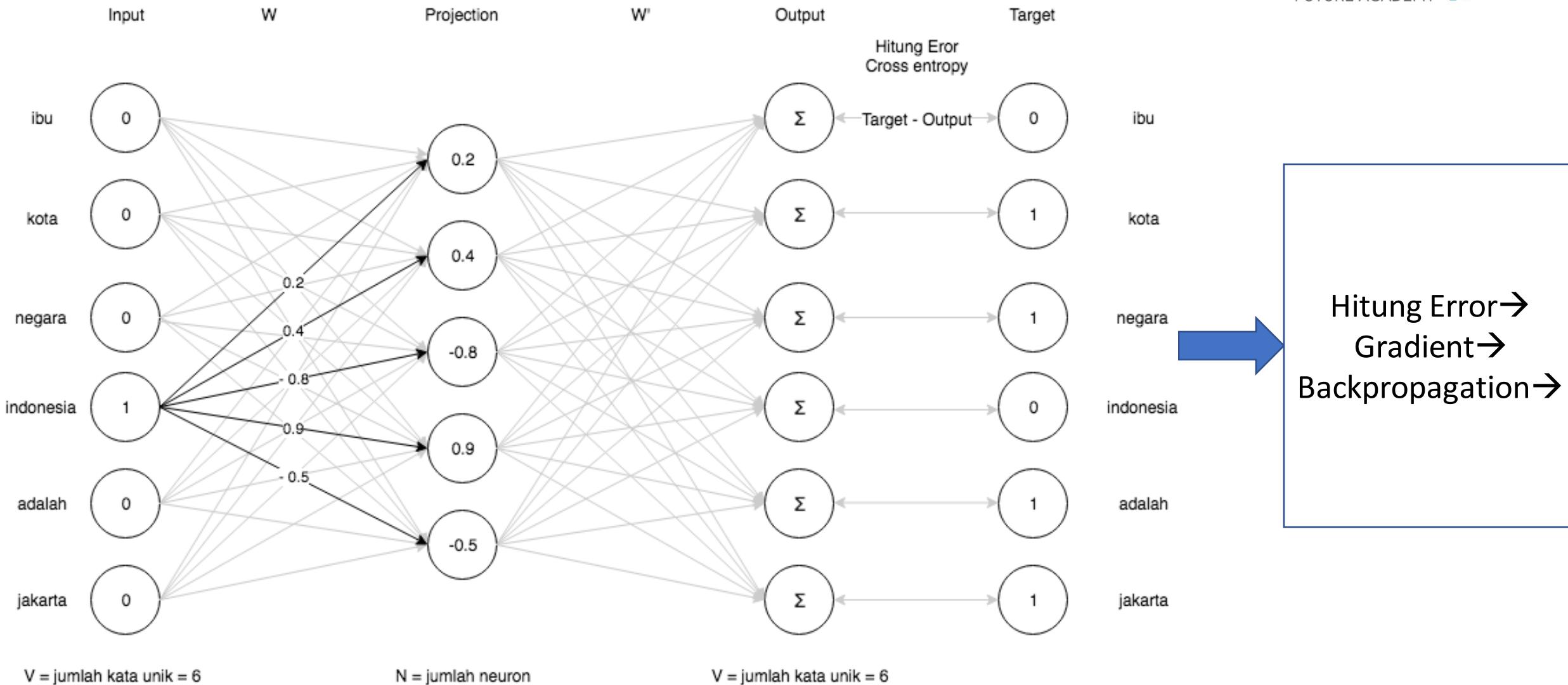
Ibu kota Negara Indonesia adalah Jakarta

target input target

Input : Satu Kata
Output : Konteks kata di sekitar input



Word2Vec - Skipgram



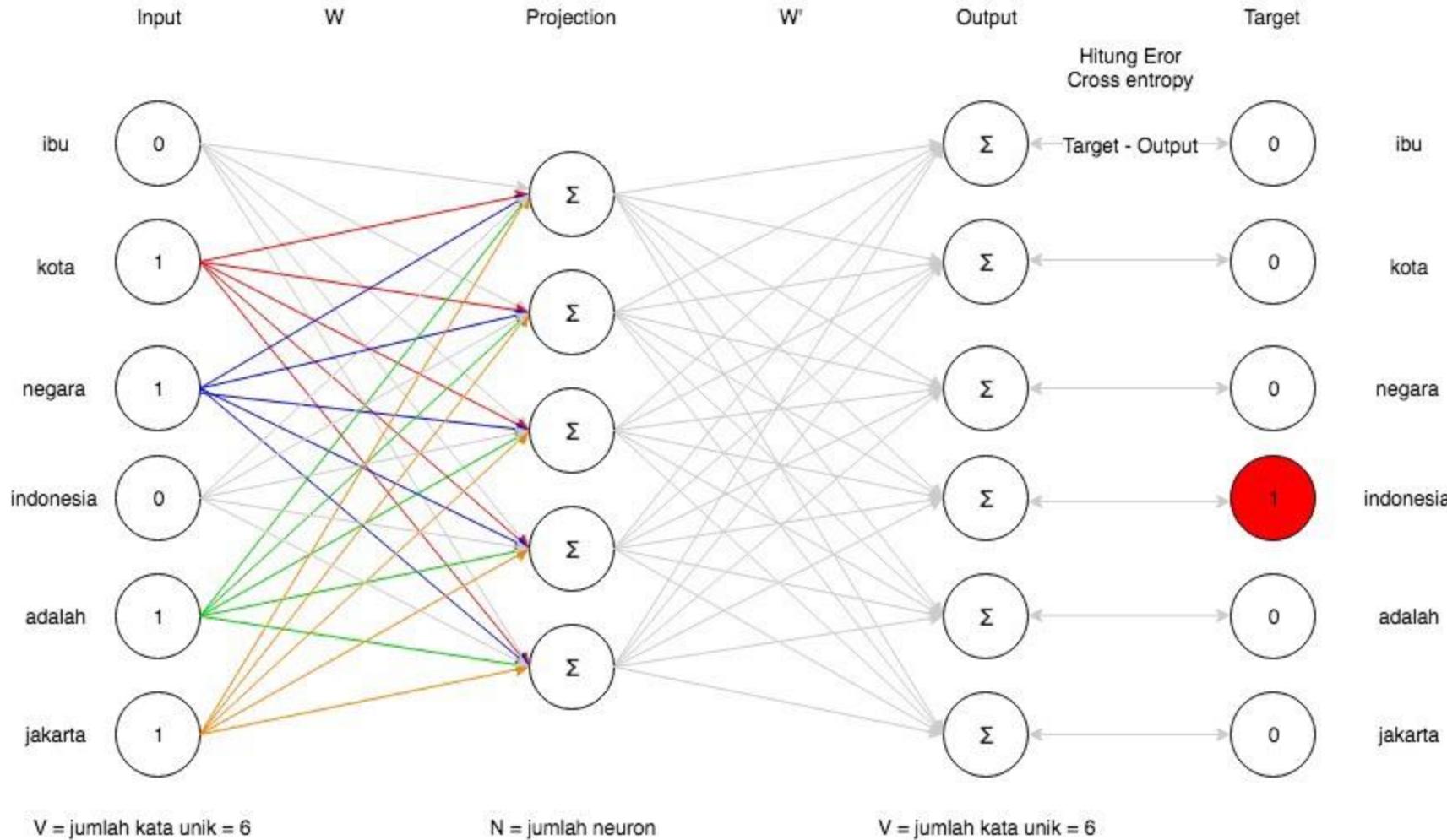
Word2Vec - CBOW

CBOW atau Continuous Bag of words

- Melakukan prediksi suatu kata dengan diberikannya kata-kata disekitarnya



Word2Vec - CBOW

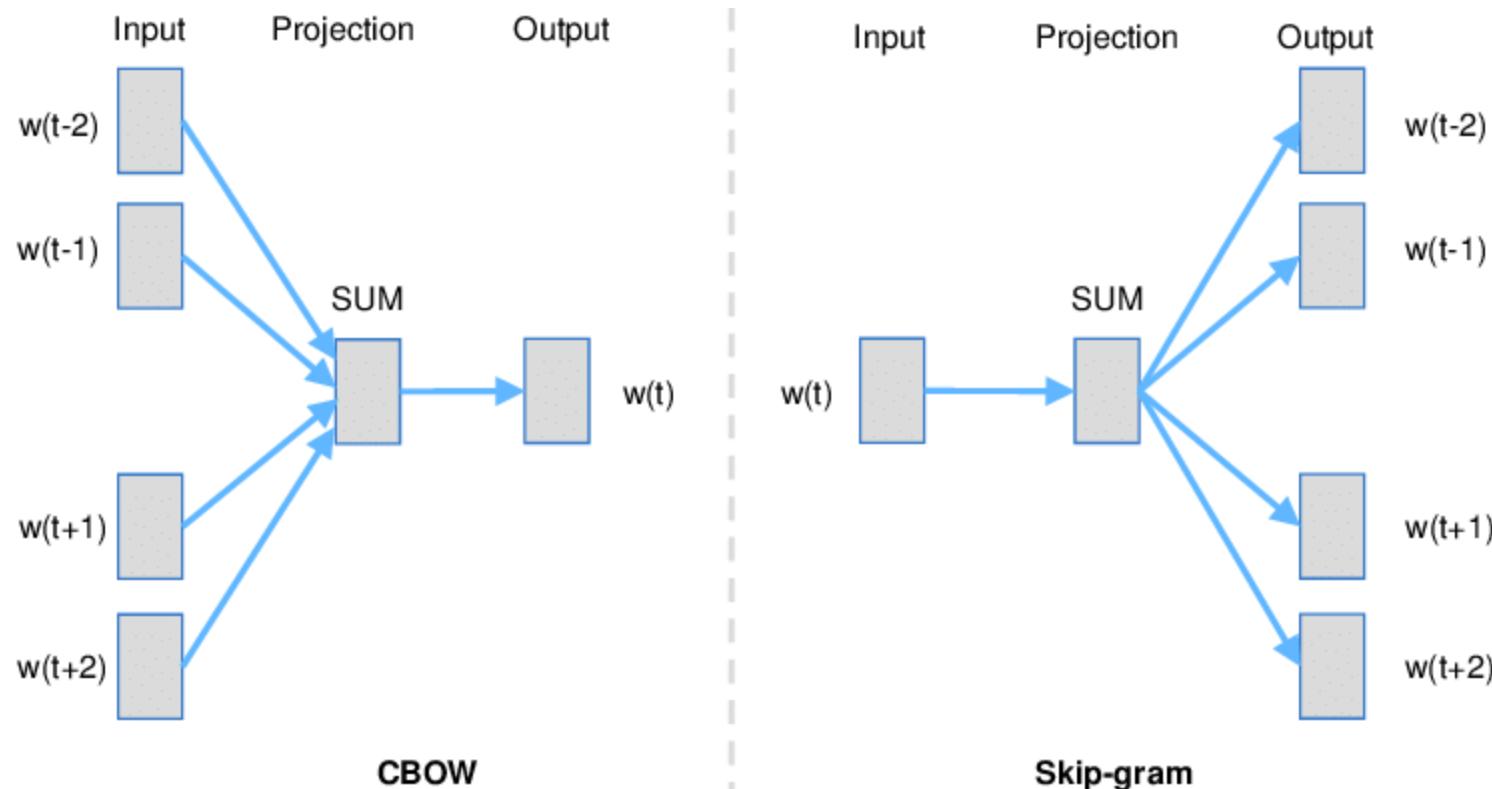


→ Hitung Error → Gradient → Backpropagation →

Word2Vec – Skipgram vs CBOW

Skipgram: memprediksi semantik dari kata pada suatu kalimat

CBOW : memprediksi satu kata dari semantik (input) pada suatu kalimat



Kapan menggunakan ?

- Jika dataset sedikit maka skipgram dapat dijadikan pilihan.
- CBOW memiliki waktu training yang lebih singkat.
- Selain itu, jika memiliki kata kata berulang Banyak seperti kata 'saya' yang berulang di Berbagai konteks kalimat. CBOW menjadi pilihan, sebaliknya untuk Skipgram

Sumber Gambar : <https://www.researchgate.net/profile/Daniel-Braun-6/publication/326588219/figure/fig1/AS:652185784295425@1532504616288/Continuous-Bag-of-words-CBOW-CB-and-Skip-gram-SG-training-model-illustrations.png>

Word2Vec

Misalkan terdapat 300 dimensi dan 10.000 total kata, berapa banyak parameter yang perlu diupdate (dihitung)?

Terdapat total 3 juta parameter weight yang harus diupdate

Mikolov dkk, menyadari bahwa terdapat banyak parameter yang akan diupdate, maka dari itu pada papernya :

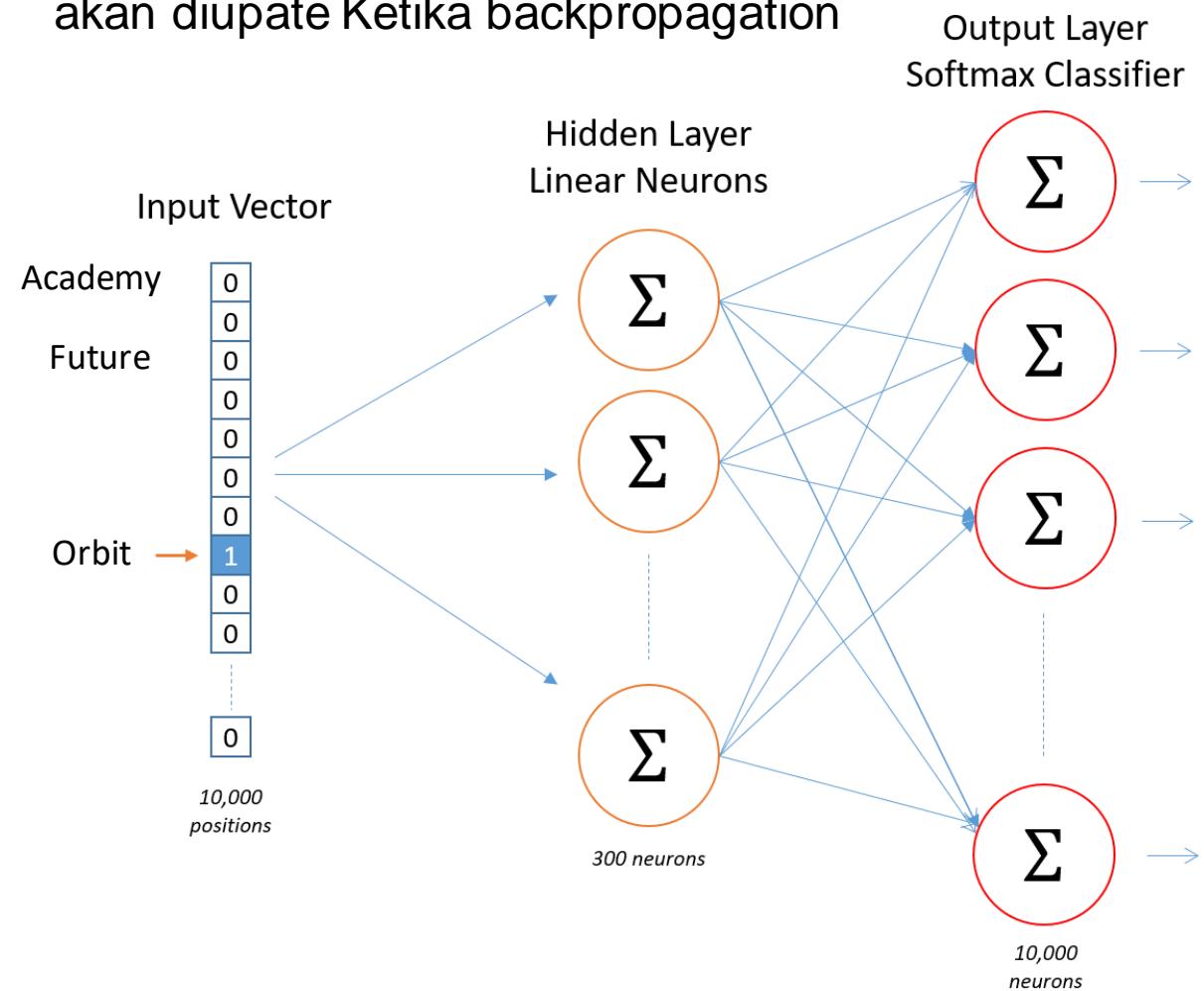
Distributed Representations of Words and Phrases and their Compositionality (neurips.cc)

Memperkenalkan operasi negative sampling

Negative Sampling ialah mekanisme untuk mengurangi jumlah parameter yang akan diupdate.

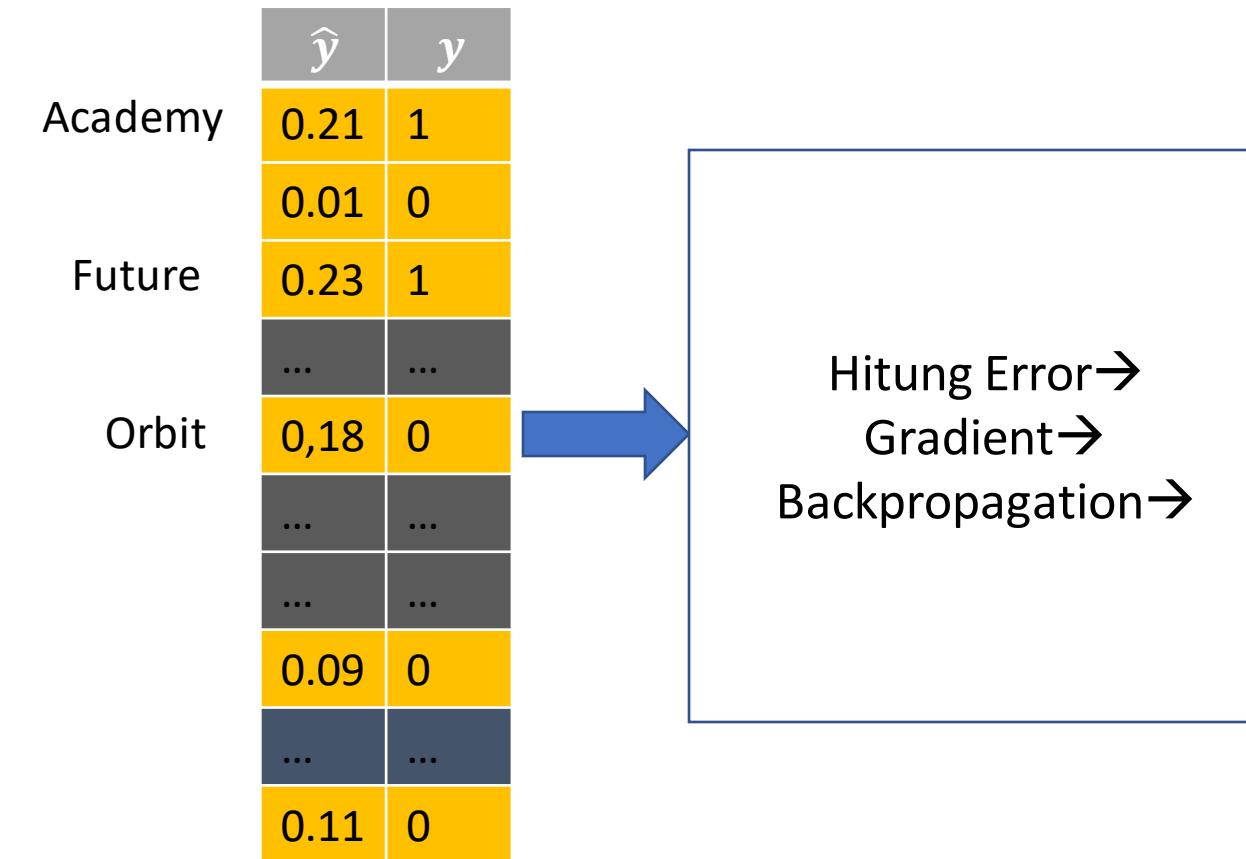
Word2Vec – Negative Sampling

Negative Sampling digunakan untuk mengurangi jumlah parameter yang akan diupdate Ketika backpropagation

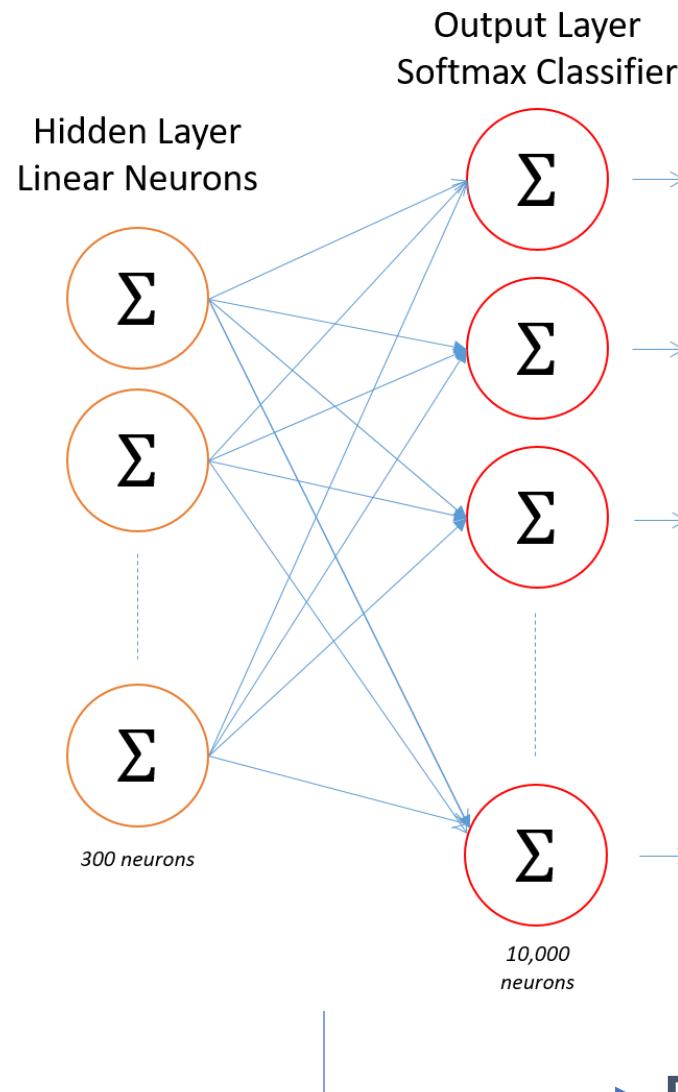


Bagaimana Negative Sampling Bekerja ?

Negative sampling bekerja dengan memperhatikan target yang bernilai 1, dan mengabaikan target bernilai 0 dengan rate tertentu.



Word2Vec – Embedding Matrix/table



Setelah training akan diperoleh embedding yang dapat berupa tabel berikut

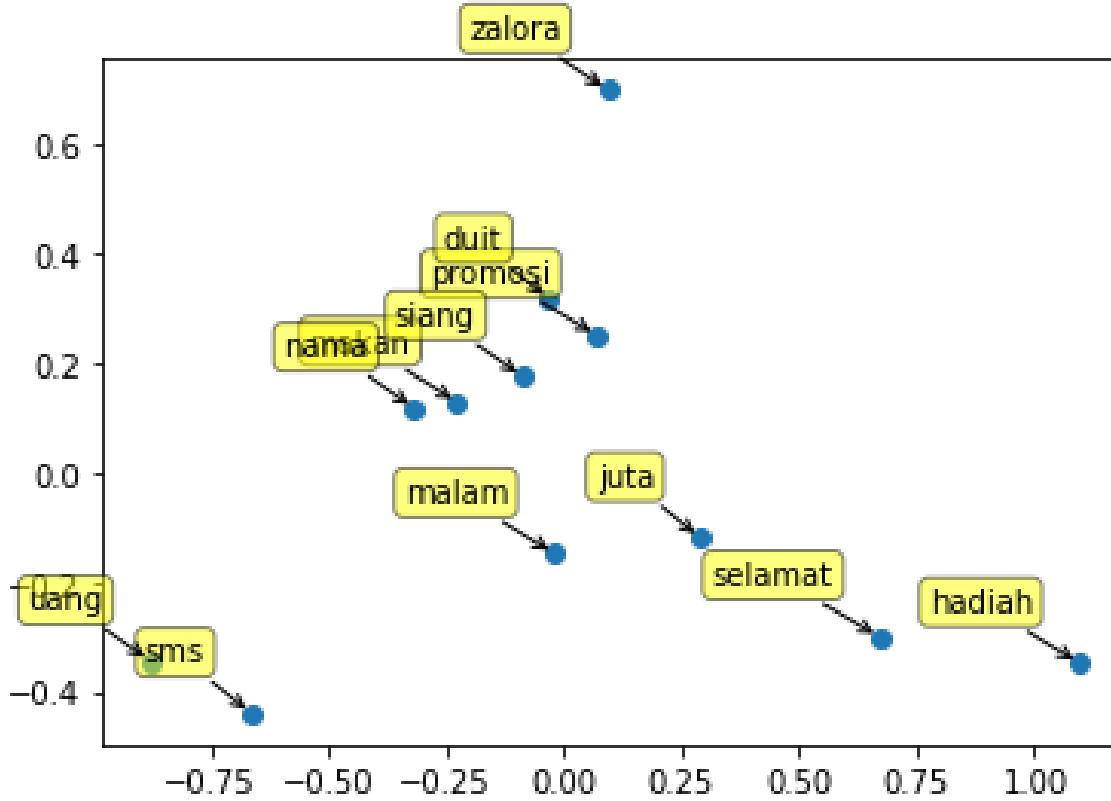
| Indeks | Kata | 5 dimensi | | | | |
|--------|-------|-----------|------|------|------|------|
| | | 0.11 | 0.23 | 0.3 | 0.44 | 0.11 |
| 1 | Abah | 0.11 | 0.23 | 0.3 | 0.44 | 0.11 |
| 2 | Abang | 0.19 | 0.22 | 0.13 | 0.78 | 0.12 |
| ... | ... | ... | ... | ... | ... | ... |
| 900 | Orbit | 0.11 | 0.21 | 0.43 | 0.88 | 0.34 |
| ... | ... | ... | ... | ... | ... | ... |
| 1000 | Zulu | 0.9 | 0.11 | 0.22 | 0.42 | 0.19 |

Maka embedding kata 'orbit' adalah : [0.11, 0.21, 0.43, 0.88, 0.34]

Dimensi = banyaknya embedding = neuron pada hidden layer

Word2Vec – Semantic (Analogy)

Setelah proses Latihan, selain mendapatkan table embedding sebelumnya, kita juga dapat merepresentasikan embedding dalam bidang 2 dimensi dan memperoleh analogi di dalamnya



Sebelumnya kita punya embedding dengan 50 dimensi, namun sebenarnya hasil embedding dapat direpresentasikan pada bidang 2 dimensi seperti pada gambar

Menurutmu bagaimana cara merepresentasikannya ke bidang 2 dimensi ?

FastText

- FastText adalah word embedding yang dihasilkan oleh Facebook
- Di awal kita sempat menyinggung Out of Vocabulary (OOV)
- Fasttext dapat mengatasi OOV
- Cara kerjanya sama dengan Word2Vec. namun kata di-embed dengan special karakter tertentu pada tingkat karakter (n-gram)

Dalam publikasinya sendiri n yang digunakan ialah $3 \leq n \leq 6$. Setiap n dilakukan embedding satu persatu dan keseluruhan n-gram dijumlahkan dan dihasilkan word embedding.
Jika tertarik berikut paper original dari FastText

[1607.04606.pdf \(arxiv.org\)](https://arxiv.org/pdf/1607.04606.pdf)

- Misalkan kata ‘orbit’ dengan $n=3$, maka embeddingnya direpresentasikan oleh:

| | | | | |
|-----|-----|-----|-----|-----|
| <or | orb | rbi | bit | it> |
|-----|-----|-----|-----|-----|

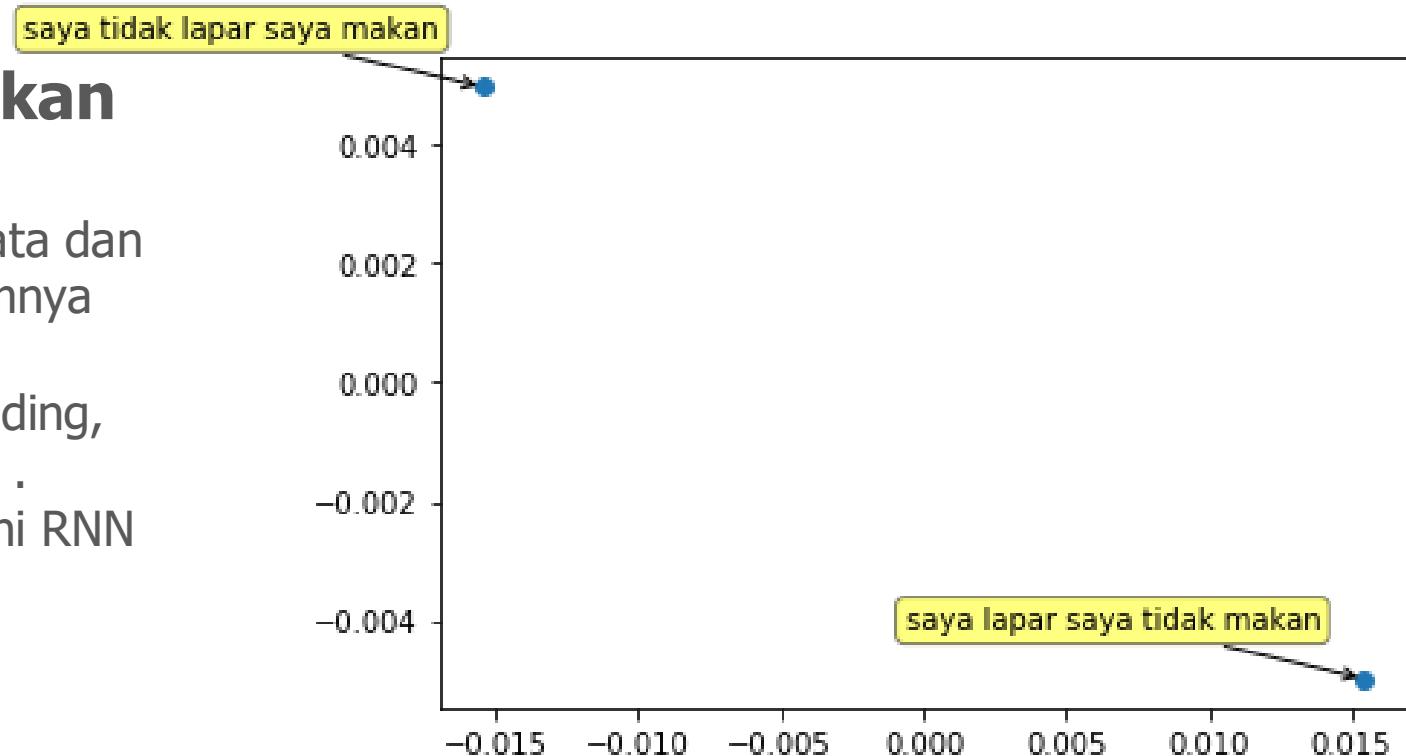
Simbol ‘<’ sebagai penanda awal kata, sebaliknya dengan ‘>’

Hal ini agar dapat membedakan embedding di awal atau tengah kata

FastText

Bagaimana FastText menyelesaikan kasus OOV ?

- Fasttext akan merekonstruksi kembali setiap kata dan menyusun ulang dari n-gram karakter sebelumnya yang dibangun
- Fasttext juga dapat dijadikan sentence embedding, namun sebenarnya pendekatan ini tidak stabil .
- Terdapat pendekatan lain yang lebih baik yakni RNN dan LSTM yang akan dipelajari selanjutnya



Contoh kontruksi dari FastText:

Misalkan di kamus sudah terdapat kata "parasut" dan "layangan"

Sedangkan pada data latih terdapat kata baru "paralayang"

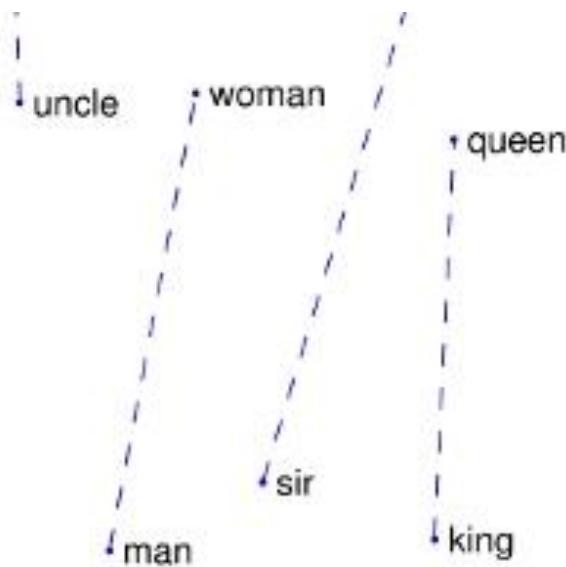
Maka kata "paralayang" dapat dikontruksi menjadi dimensi "para" dan "layang" dari kata "parasut" dan "layangan

Contoh sentence embedding dengan menggunakan FastText, terlihat kalimat yang disinggung di awal PPT yakni: 'saya tidak lapar, saya makan' dan 'saya lapar, saya tidak makan' Dengan fastText diperoleh representasi berbeda pada bidang 2 dimensi

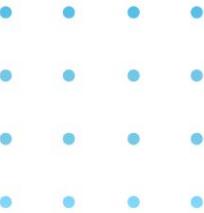
Glove

Glove adalah word embedding yang memperhatikan kemunculan kata pada korpus

Jika tertarik anda dapat mempelajari lebih lanjut pada:
<https://nlp.stanford.edu/pubs/glove.pdf>



[GloVe: Global Vectors for Word Representation \(stanford.edu\)](https://nlp.stanford.edu/pubs/glove.pdf)



03 RNN

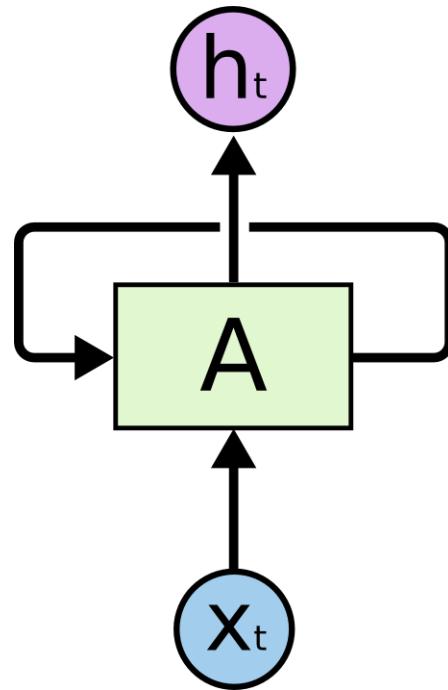
- Definisi dari Recurrent Neural Network
- Ide dari penggunaan RNN



Recurrent Neural Network

Guna memperoleh sentence embedding yang lebih baik. Kita dapat menggunakan RNN.

Intuisinya ialah: kalimat terbentuk dari kata-kata, sehingga dengan memperhatikan suatu kata dengan kata lain akan membentuk satu gambaran utuh tentang kalimat



- RNN memanfaatkan looping setiap kata.
- Input x_t berupa vector (dapat berupa word embedding dari word2vec dst)
- A merupakan neural network
- Hasil dari neural network (output) dijadikan input pada neural network di sebelahnya dan juga menjadi output dari neural network itu sendiri h_t

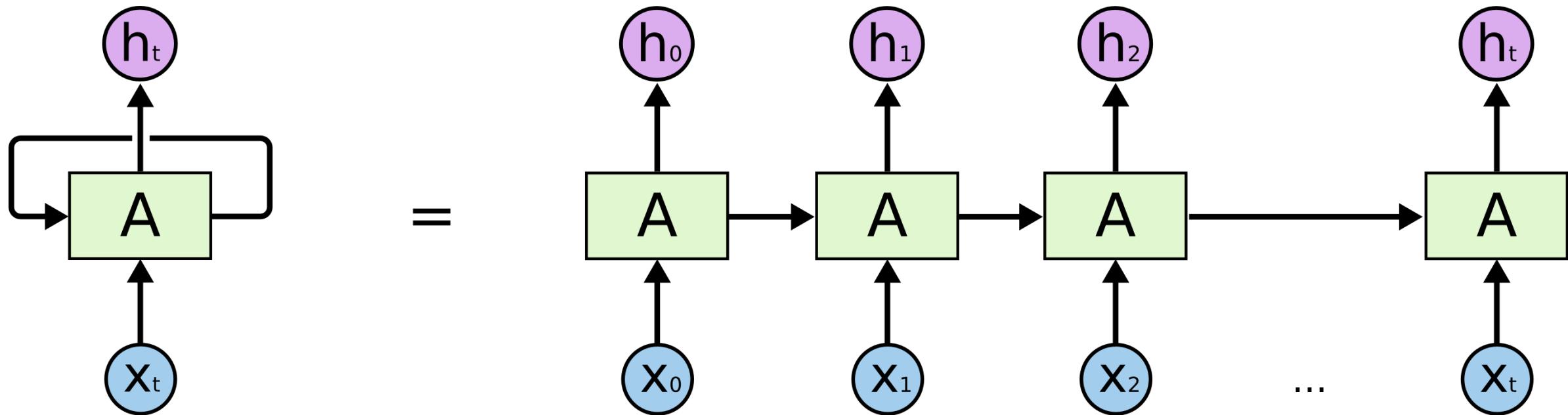
Recurrent Neural Network

Jelasnya sebagai berikut:

Terdapat x_0, x_1, \dots sebagai input (vector), output dari A pertama dijadikan input pada A kedua.

Selain itu h_0, h_1, \dots merupakan output dari A.

A sendiri disebut sebagai cell



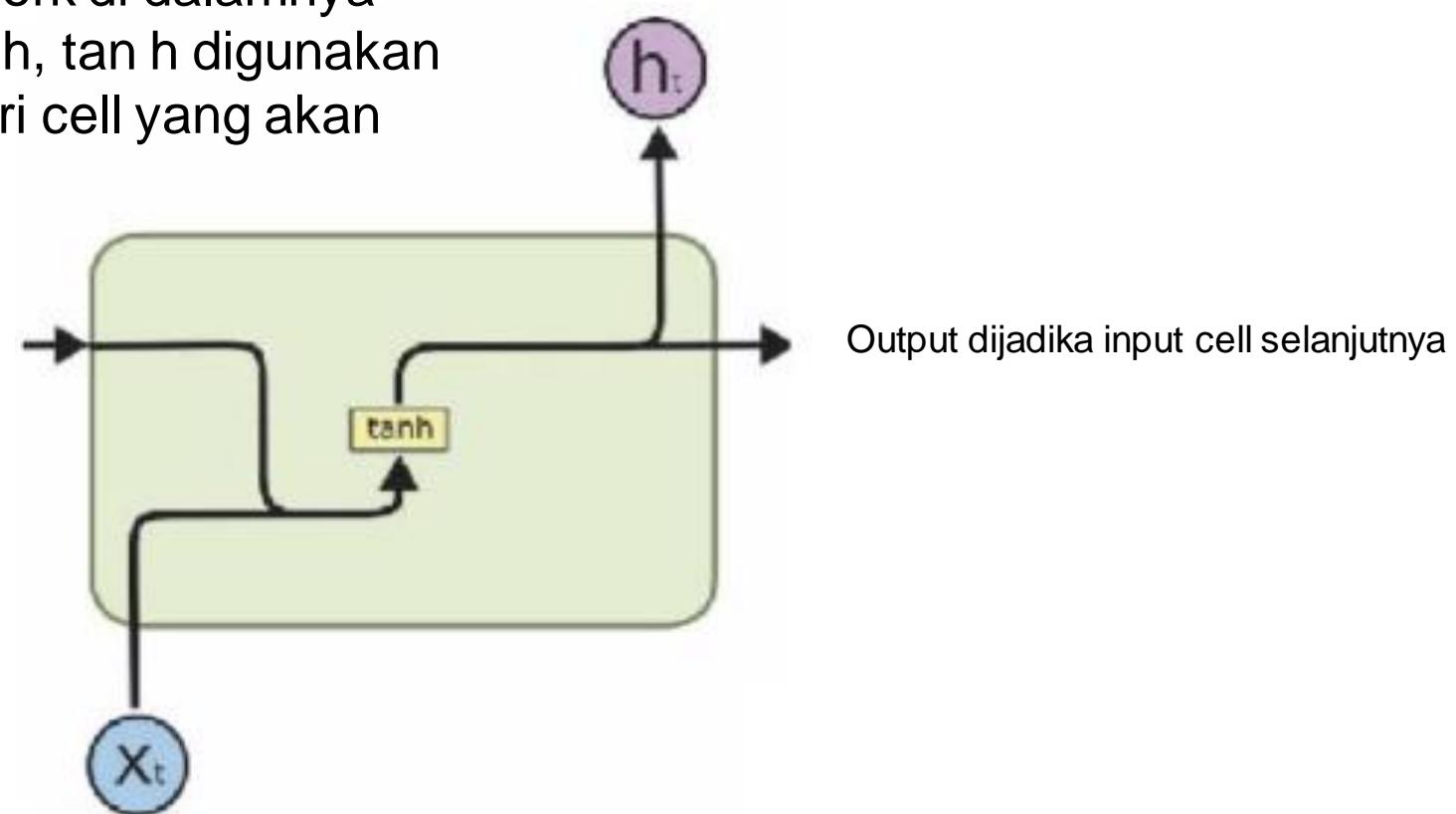
Recurrent Neural Network

Apa yang terdapat pada A ?

Adapun yang terjadi pada A atau disebut cell ialah:

Terdapat operasi konkatenasi dari x_t (input ke-t) dan output dari cell sebelumnya yakni h_{t-1}

Selain itu terdapat neural network di dalamnya dengan activation function tan h, tan h digunakan untuk menyeleksi informasi dari cell yang akan dijadikan output

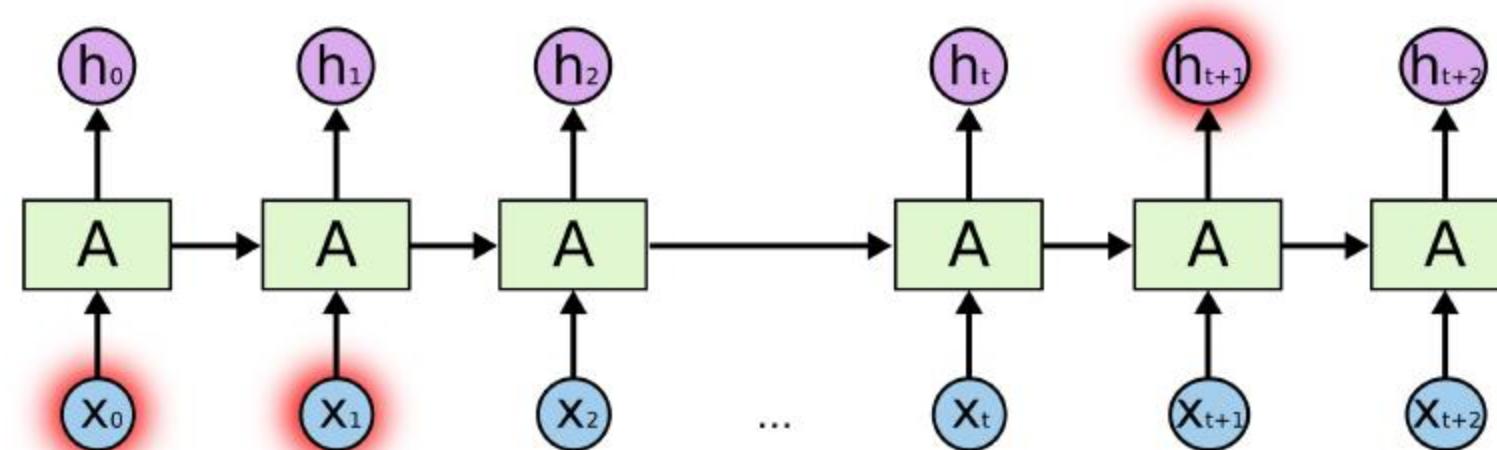


Recurrent Neural Network

Long Dependency problem

Namun, RNN memiliki kendala dimana kalimat yang panjang akan membuat RNN kita lupa. Hal ini sebenarnya masuk akal, sering kali kita sebagai manusia lupa akan bacaan kita sebelumnya.

Kita dapat mengulang bacaan tersebut jika lupa. Namun **bagaimana dengan AI ?**

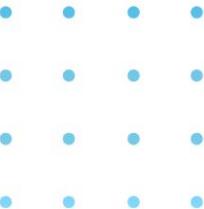


Contoh dari *long dependency problem* ialah:

Budi juara kelas dan ia senang, sedangkan Sarah tidak menjadi juara kelas, ia bersedih

RNN akan mengingat kata ia merujuk pada Budi dan Sarah. Padahal apabila terdapat kata "ia" lagi di kalimat selanjutnya kata 'ia' seharusnya merujuk pada "Sarah".

Karena RNN mengingat 'Budi' dan 'Sarah' ada saatnya RNN lupa 'ia' selanjutnya merujuk kepada siapa



04 LSTM

- Definisi dari Long Short-Term Memory
- Ide penggunaan LSTM



Long Short-Term Memory (LSTM)

- Guna mengatasi long dependency problem, kita dapat menggunakan LSTM. Pada LSTM kita hanya memperhatikan informasi yang relevan dan melupakan (filter) kata-kata yang tidak relevan.
- Misalnya kata “Budi juara kelas dan ia senang, sedangkan Sarah tidak menjadi juara kelas, ia bersedih”

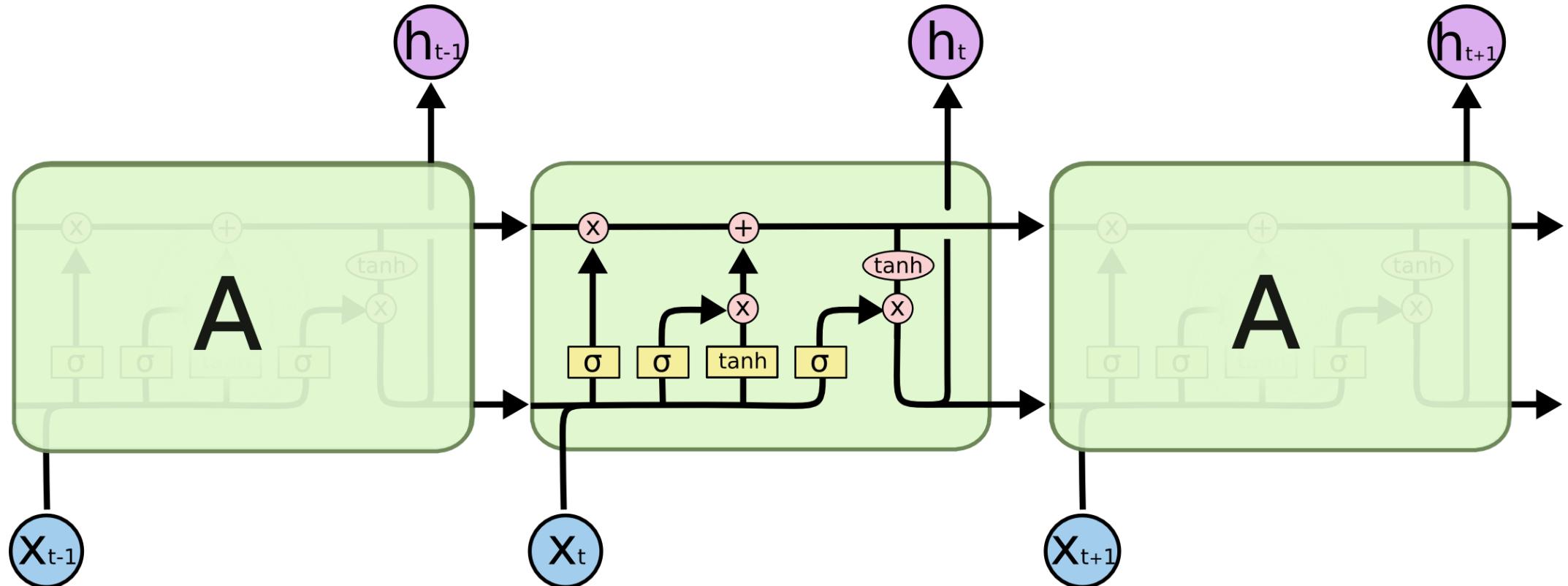
Sarah akan diingat dan kata ‘ia’ hanya merujuk ke ‘Sarah’. Kata “Budi” akan difilter dan dilupakan pada cell-cell selanjutnya

- LSTM adalah modifikasi dari RNN

Long Short-Term Memory (LSTM)

Pada LSTM, terdapat istilah cell state yang akan diteruskan ke setiap cell.

Terjadi pula operasi Hadamard product (elemen-wise product) dan penjumlahan untuk melupakan (filter) kata-kata tertentu dan memberikan informasi baru yang dianggap penting



Long Short-Term Memory (LSTM)

Forget Gate

Dengan activation sigmoid (0-1)

Terjadi operasi perkalian antar elemen

Terhadap cell state dari cell sebelumnya.

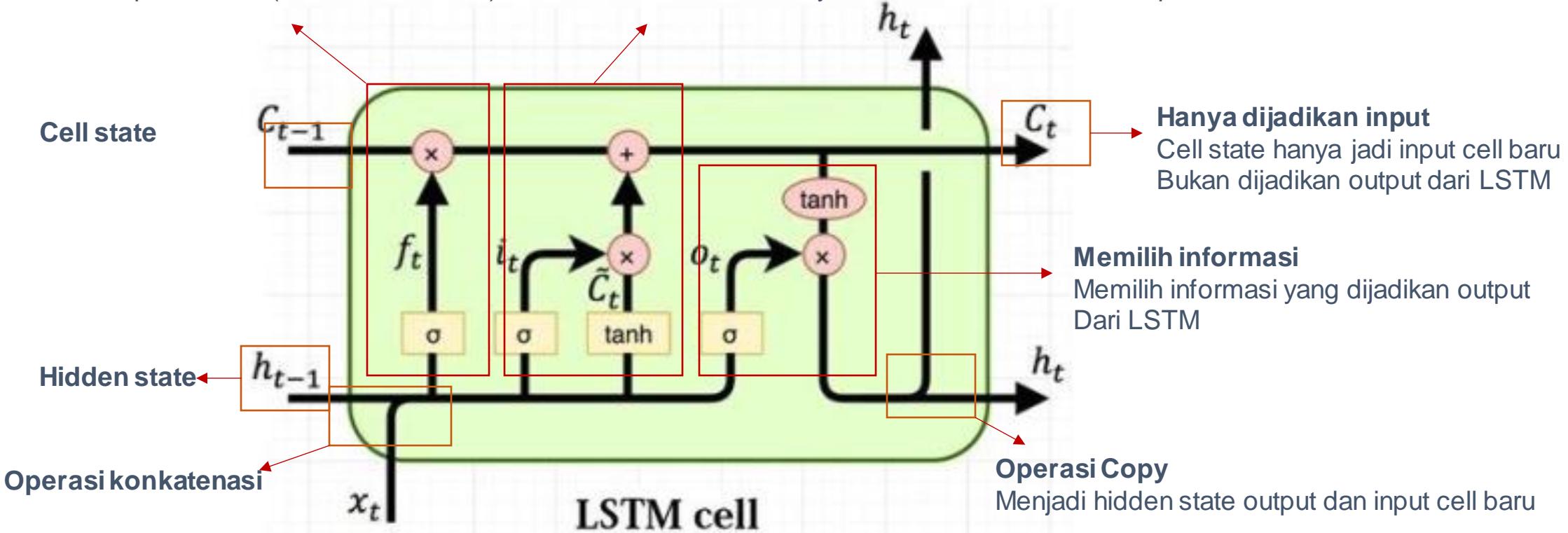
Apabila dikali 0 maka lupakan kata (informasi tersebut)

Input Gate

terdapat activation sigmoid (0-1)

Dikatakan input gate karena memberikan informasi baru kepada cell state.

Operasi yang sama pada forget gate yakni terdapat operasi perkalian antar elemen, namun setelahnya hasil ditambahkan terhadap cell state

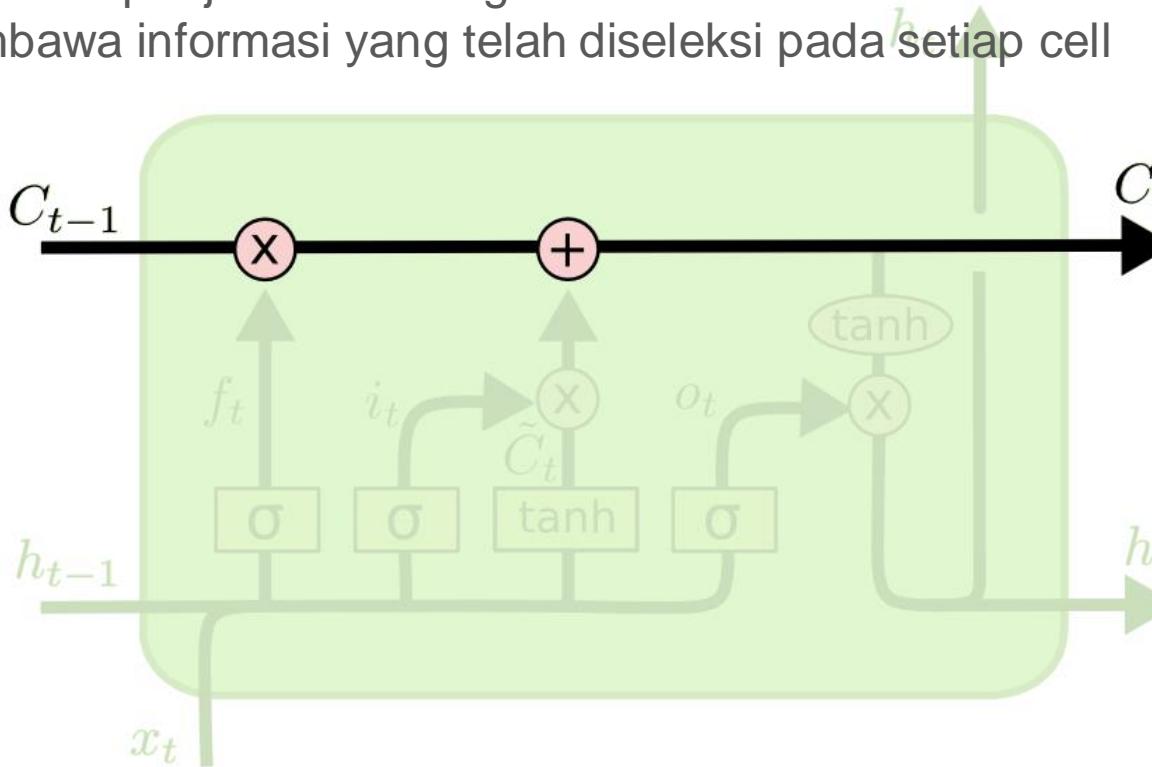


Long Short-Term Memory (LSTM)

Ide Utama dari LSTM ialah **cell state**, dimana terdapat mekanisme mengingat informasi yang penting atau men-filter atau melupakan informasi yang tidak relevan lagi dengan operasi element-wise product .

Selain itu penambahan operasi penjumlahan berguna memberikan informasi baru.

Cell State akan terus membawa informasi yang telah diseleksi pada setiap cell

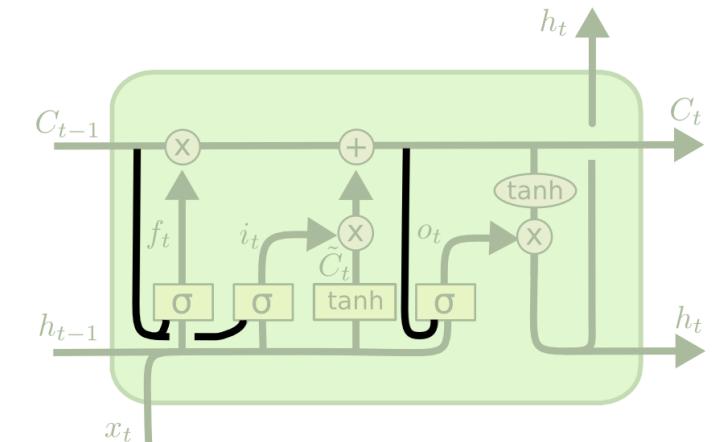
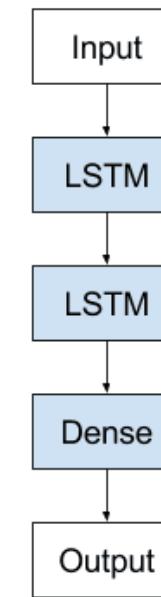
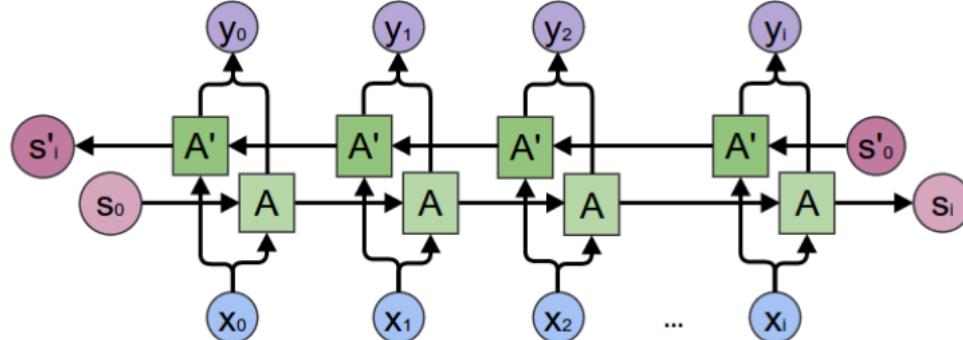


Simplenya kita mengajarkan LSTM untuk mengingat-ingat atau melupakan informasi yang relevan. Berbeda dengan RNN dimana seluruh informasi kita suruh untuk selalu dibawa untuk setiap cell yang malah mengarah pada long dependency problem

Long Short-Term Memory (LSTM)

Walaupun LSTM dapat menyelesaikan long dependency problem dan vanishing gradient problem (Ketika update weight, gradient menjadi 0)

LSTM juga dapat di-improve misalnya dengan menggunakan Bidirectional LSTM (bi-LSTM) atau melakukan Penumpukan LSTM (Stacking LSTM)



Terdapat banyak modifikasi dari LSTM lain, seperti menggunakan 'peep hole'
 Jika tertarik dapat mempelajarinya di:[Understanding LSTM Networks -- colah's blog](#)

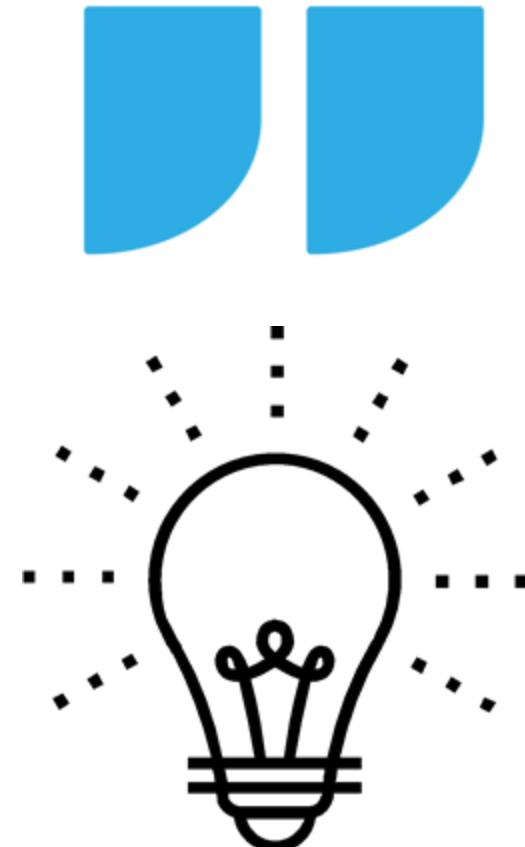


05 KESIMPULAN

- Ringkasan
- Kuis

Ringkasan

1. Word embedding menjawab kekurangan dari pendekatan sebelumnya, yakni data sparse, low semantics dan OOV (khususnya FastText)
2. Word embedding diperoleh dengan melatih shallow neural network (satu layer), weight yang dihasilkan dijadikan word embedding, contoh word embedding ialah Word2Vec, FastText dan Glove
3. Word2Vec dapat dilatih dengan metode CBOW dan Skipgram. Negative sampling dibutuhkan untuk mereduksi besarnya parameter
4. Word embedding hanya dapat memperoleh semantic pada level kata, dibutuhkan pendekatan yang dapat memperoleh semantic dari kalimat
5. Kalimat tersusun dari kata sehingga untuk memperoleh semantic kalimat dapat dengan melakukan pelatihan setiap kata pada kalimat
6. RNN adalah pendekatan yang dapat memperoleh semantic kalimat, namun ia memiliki kelemahan yakni Long Dependecy Problem
7. Long Dependency Problem dapat diselesaikan dengan LSTM.
8. LSTM bekerja dengan memperhatikan informasi yang relevan dan melupakan informasi yang tidak relevan lagi



Kuis

Pertanyaan

Pada saat training word2vec, inisialisasi weight bersifat random ?

- A. Benar
- B. Salah



Kuis

Pertanyaan

Pada saat training word2vec, inisialisasi weight bersifat random ?

- A. Benar
- B. Salah

Jawaban: A



Kuis

Pertanyaan

Out of vocabulary dapat diselesaikan dengan pendekatan ?

- A. BM25
- B. FastText
- C. Word2Vec
- D. TF-IDF



Kuis

Pertanyaan

Out of vocabulary dapat diselesaikan dengan pendekatan ?

- A. BM25
- B. FastText**
- C. Word2Vec
- D. TF-IDF



Jawaban: B



TERIMA KASIH

Orbit Future Academy

PT Orbit Ventura Indonesia
Center of Excellence (Jakarta Selatan)
Gedung Veteran RI, Lt.15
Unit Z15-002, Plaza Semanggi
Jl. Jenderal Sudirman Kav.50, Jakarta
12930, Indonesia

- Jakarta Selatan/Pusat
- Jakarta Barat/BSD
- Kota Bandung
- Kab. Bandung
- Jawa Barat

Hubungi Kami

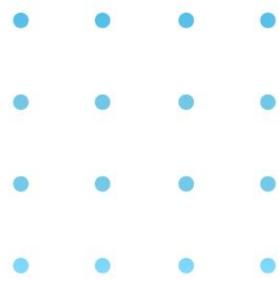
Director of Sales & Partnership
ira@orbitventura.com
+62 858-9187-7388

Social Media

-  [Orbit Future Academy](#)
-  [@OrbitFutureAcademyIn1](#)
-  [OrbitFutureAcademy](#)
-  [Orbit Future Academy](#)



AI Mastery Course



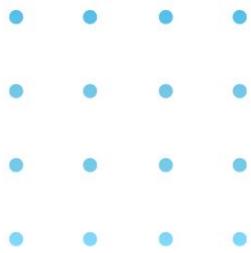
Module 10

Natural Language Processing (NLP)

Section

Transfer Learning for NLP –
Transformer & BERT





Learning Objectives

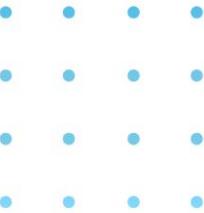
Di akhir modul ini, Anda akan mendapatkan:

- Memahami konsep transfer learning pada NLP.
- Memahami batasan word embedding tradisional.
- Memahami mekanisme self-attention pada arsitektur transformer.
- Memahami arsitektur dan representasi input BERT.
- Mampu melakukan fine-tuning menggunakan BERT.



Agenda

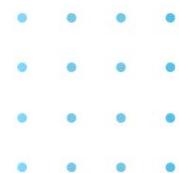
- 01 TRANSFER LEARNING PADA NLP
 - Konsep umum transfer learning
 - Jenis transfer learning NLP
 - Sequential transfer learning
- 02 FROM WORDS TO WORDS-IN-CONTEXT
 - Batasan Word2Vec
 - Words-in-context
- 03 TRANSFORMER
 - Arsitektur Transformer
 - Mekanisme self-attention
- 04 BERT
 - Arsitektur BERT
 - Representasi input BERT
 - Fine-Tuning BERT
- 05 KESIMPULAN
 - Ringkasan
 - Kuis



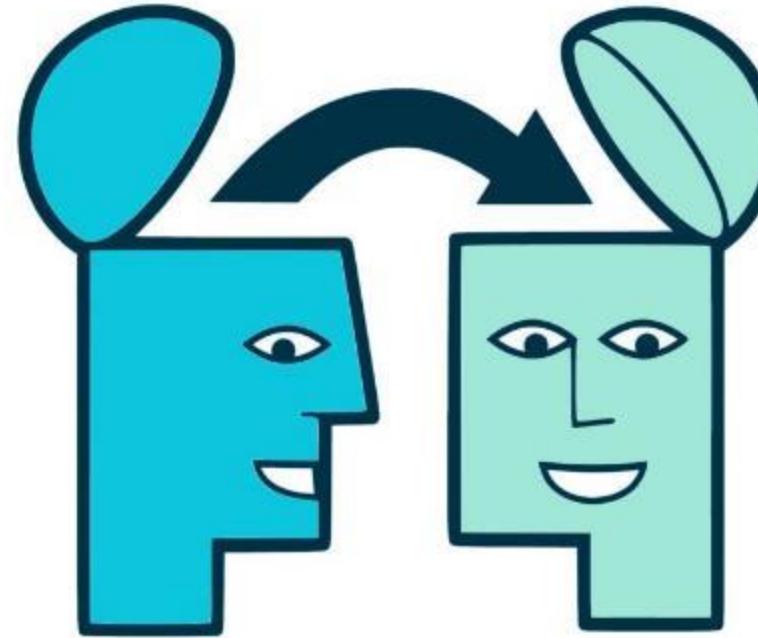
01

TRANSFER LEARNING PADA NLP

- Konsep umum transfer learning
- Jenis transfer learning NLP
- Sequential transfer learning



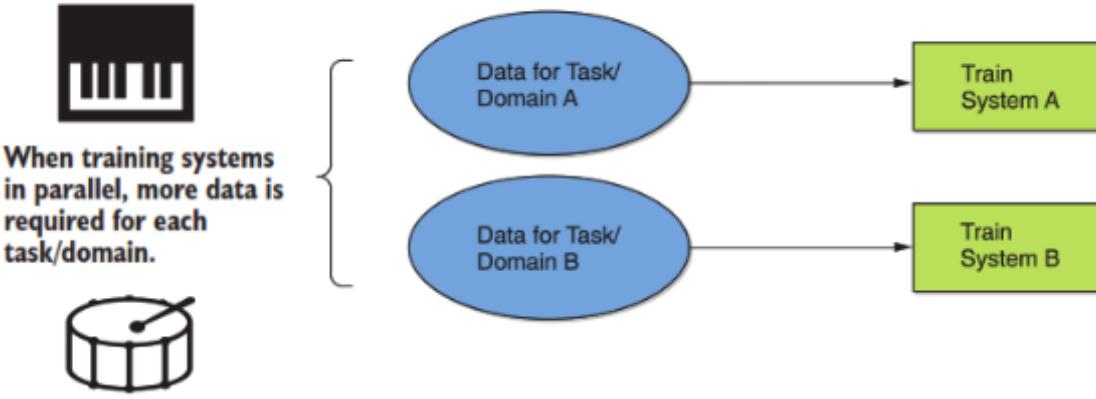
Apa itu transfer learning?



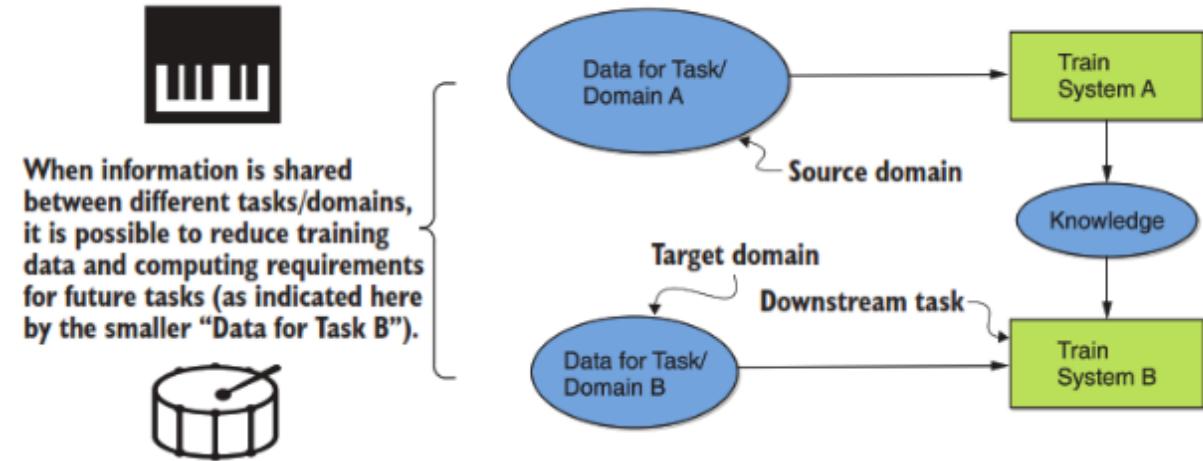
Secara umum, transfer learning adalah **melatih model** pada **tugas tertentu**, lalu model tersebut dapat **digunakan untuk tugas yang berbeda**.

Transfer Learning pada NLP

Traditional paradigm: Parallel training for different tasks/domains

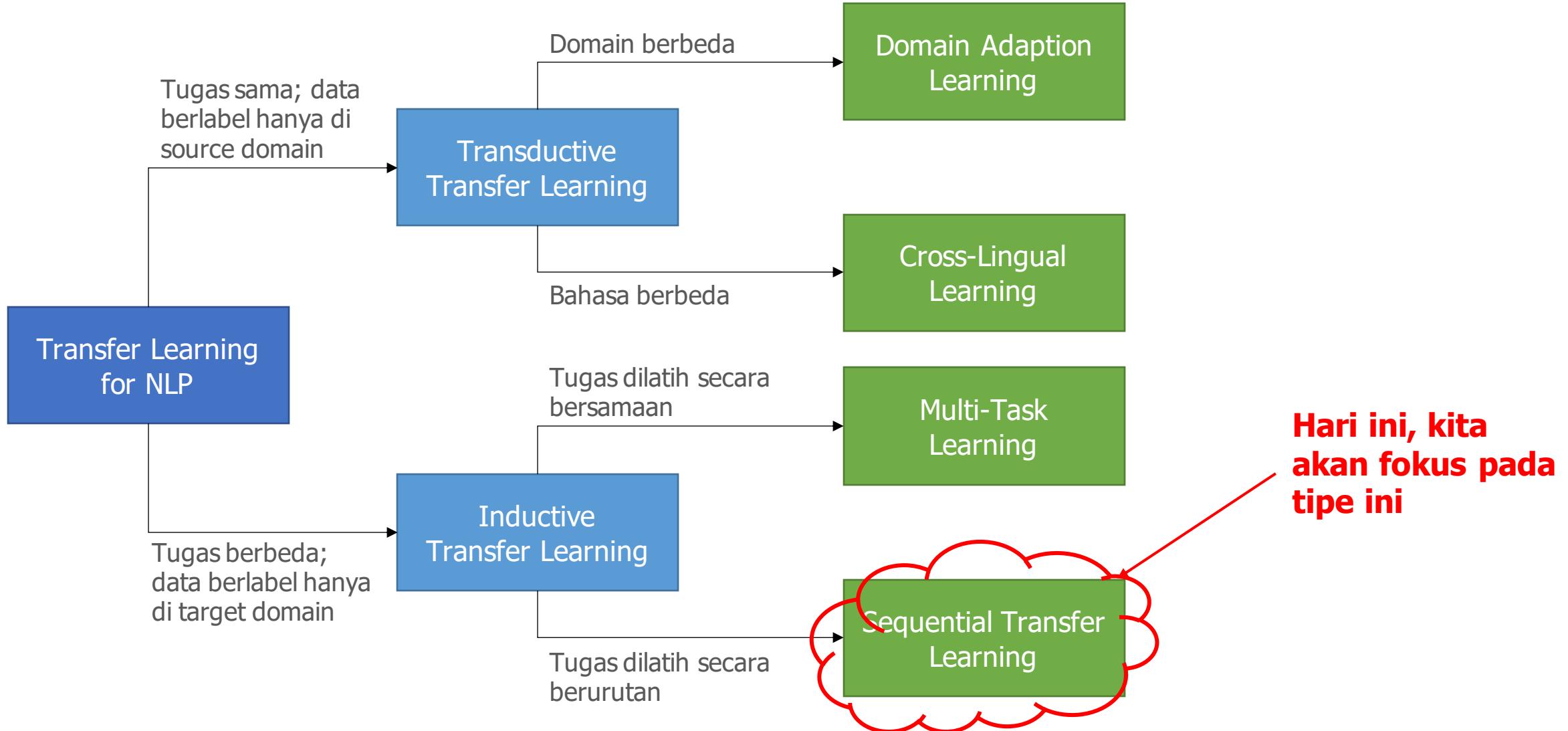


Transfer learning paradigm: Knowledge is shared between different tasks/domains.

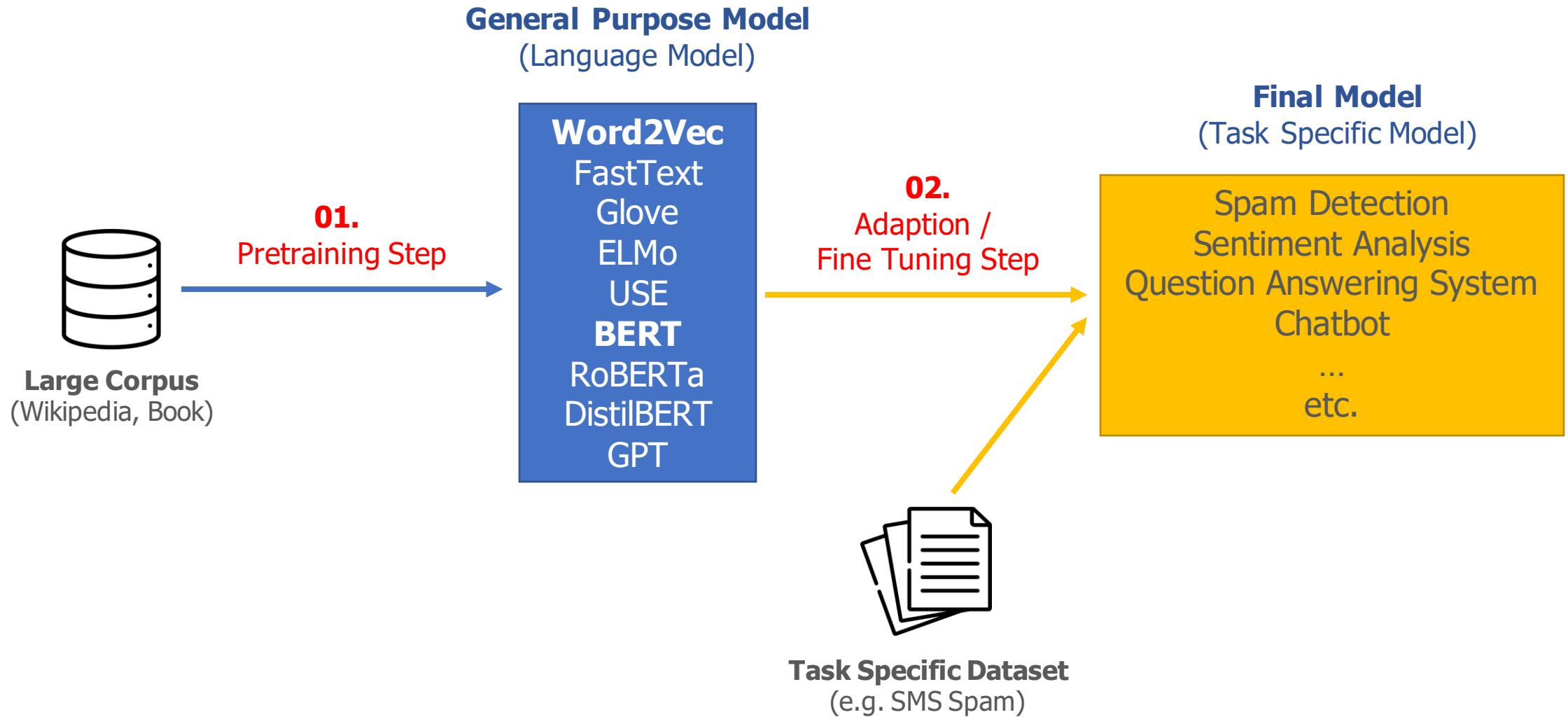


Dalam paradigma **transfer learning**, pengurangan kebutuhan data dan komputasi dapat dicapai melalui **berbagi knowledge**.

Tipe Transfer Learning NLP



Sequential Transfer Learning



Language Model Pretraining

Belajar memprediksi $P\theta(\text{teks})$ atau $P\theta(\text{teks} \mid \text{teks lain})$

Kelebihan:

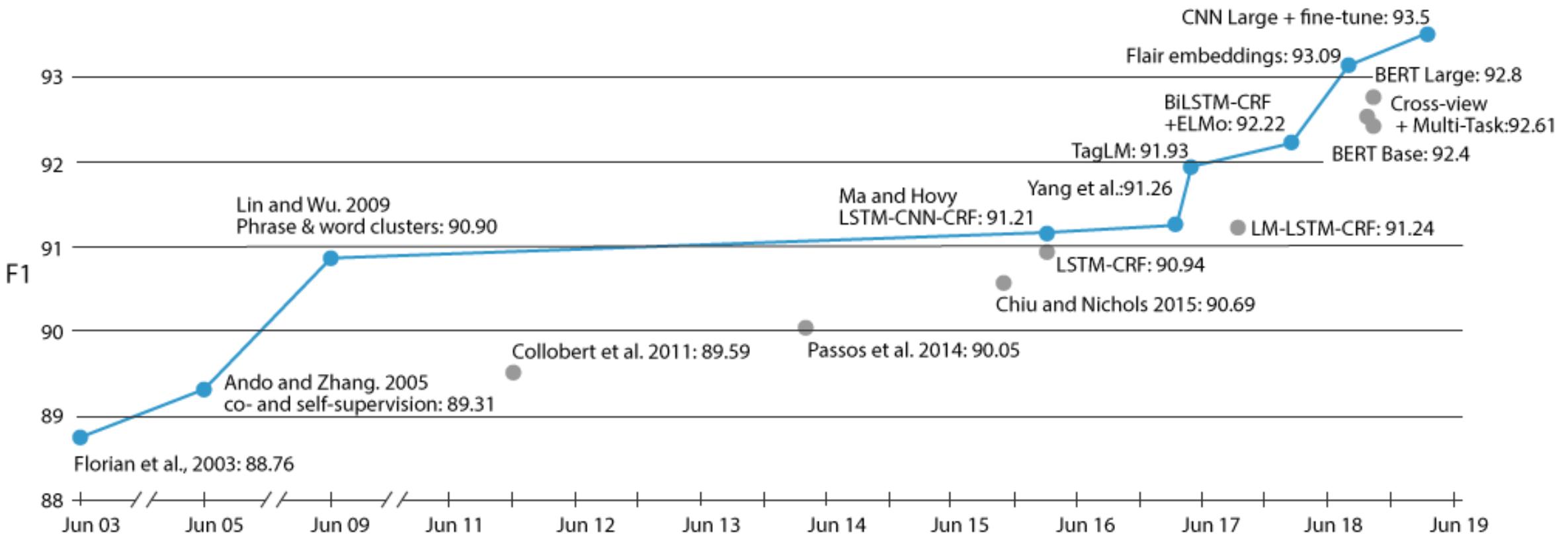
- **self-supervised** – tidak memerlukan anotasi manusia;
- Terdapat banyak bahasa memiliki sumber teks yang cukup untuk mempelajari model berkapasitas tinggi;
- **Serbaguna** – dapat digunakan untuk mempelajari representasi kalimat dan kata untuk berbagai fungsi dan tujuan.

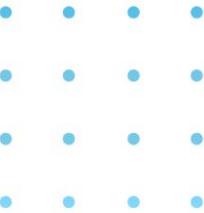
Alasan Menggunakan Transfer Learning

Batasan dari masalah NLP yang berbeda:

1. Data latih terbatas atau bahkan tidak tersedia untuk setiap tugas;
2. Membutuhkan biaya tinggi untuk membuat anotasi (pelabelan) data pelatihan.
Misalnya jika data didapatkan dari scraping;
3. Biaya komputasi yang tinggi untuk melatih model;
4. Ketersediaan sumber data untuk bahasa selain bahasa Inggris. Misalnya Bahasa daerah.

Mengapa butuh transfer learning?





02

FORM WORDS TO WORDS-IN-CONTEXT

- Batasan word2vec
- Words-in-context



Recall: Word Embedding

Teknik untuk **memetakan kata** (atau frasa) dari ruang input sparse berdimensi tinggi (BoW atau TF-IDF, dll) ke **ruang vektor padat** (dense) **berdimensi lebih rendah** (Word2Vec, FastText, GloVe, dll).

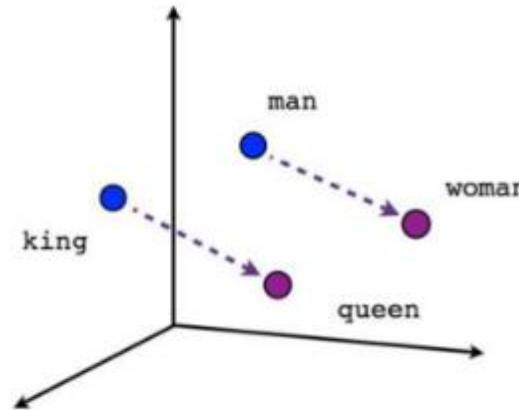
Mengapa?

Kata adalah representasi simbolis dari semantik.

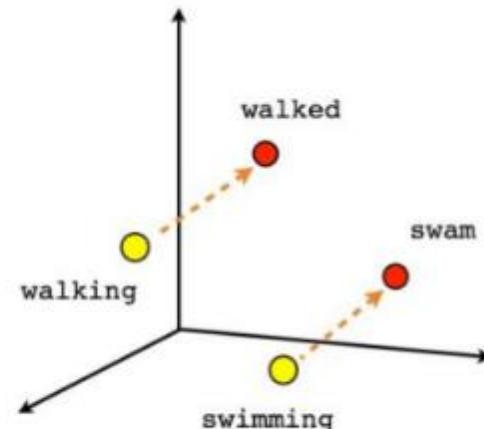
- Kata memiliki makna;
- Kata-kata dengan makna yang sama harus memiliki vektor yang hampir sama;
- Jarak antara vektor untuk konsep yang sama (contoh: sinonim) harus berdekatan.

Recall: Word2Vec

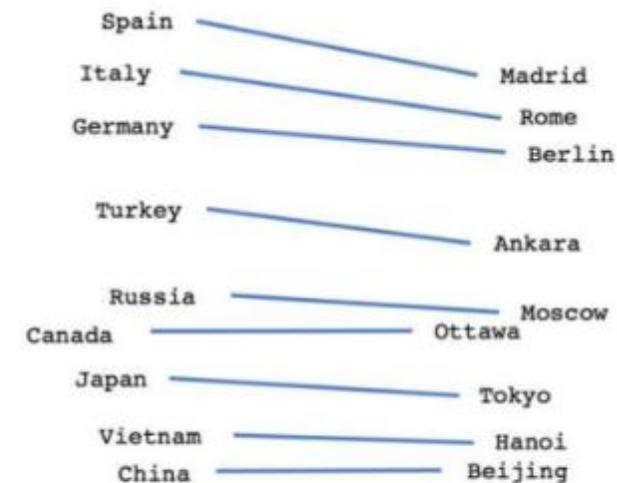
Mampu menangkap **struktur relasional** dalam kalimat, kesamaan semantik dan sintaksis, hubungan dengan antar kata.



Male-Female



Verb tense



Country-Capital

Batasan Word2Vec

- Beberapa **kata** memiliki **makna yang berbeda** (homonim) atau memiliki **lebih dari satu makna** (polisemi).
- Word2Vec diterapkan dengan cara **context-free manner**.
- **Solusi:** latih representasi kontekstual pada korpus teks berukuran besar.

Cahyo **bisa** mengerjakan soal itu



[-0.78, 3.54, 0.67]

Bisa ular itu sangat berbahaya



Fixed embedding

Cahyo **bisa** mengerjakan soal itu



[-1.75, -0.75, 1.21]

Bisa ular itu sangat berbahaya



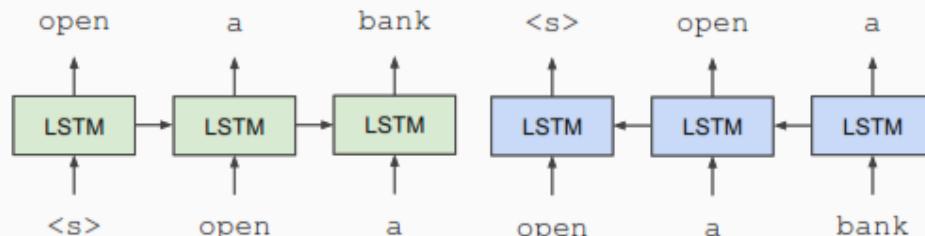
[3.65, -0.16, -0.52]

Pemahaman representasi kontekstual

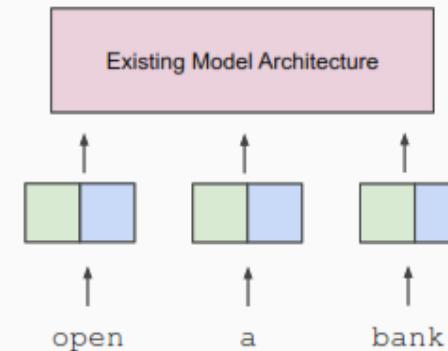
ELMo: Deep Contextualized Word Representation

Alih-alih menggunakan vektor embedding yang tetap untuk setiap kata, ELMo melihat seluruh kalimat sebelum menetapkan vektor embedding setiap kata.

Train Separate Left-to-Right and Right-to-Left LMs



Apply as “Pre-trained Embeddings”



Baca selanjutnya:

M. E. Peters et al., “Deep contextualized word representations,” arXiv.org, 2018, doi: 10.48550/arXiv.1802.05365.

<https://arxiv.org/abs/1802.05365>

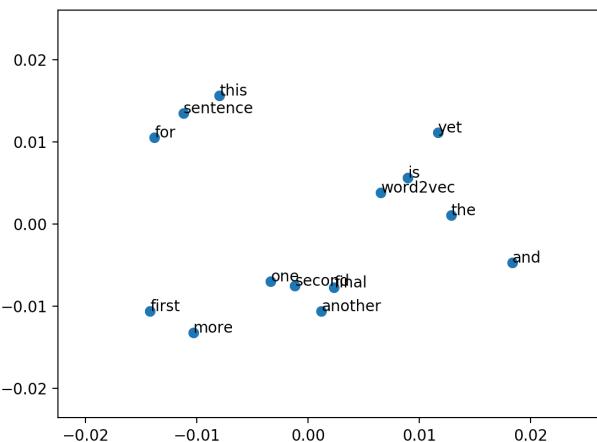


Recap: From words to words-in-context

Word Vectors

Cats = [0.2, ..., 0.3]

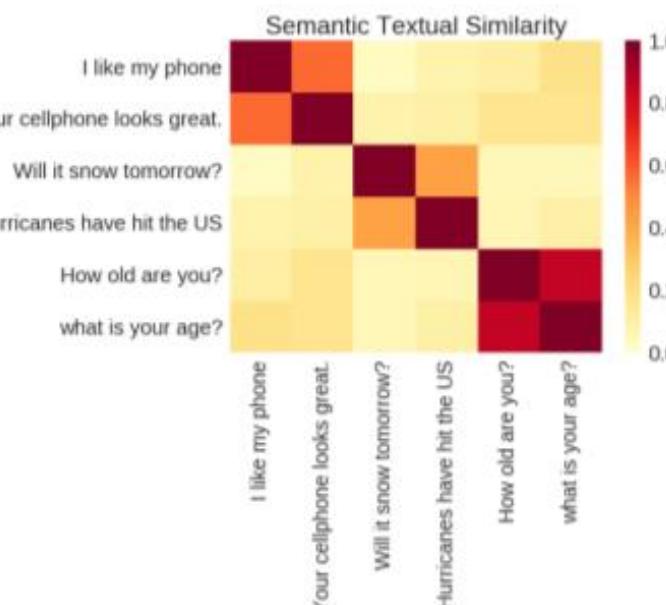
Dogs = [0.3, -0.5]



Sentence / Doc Vectors

We have two Cats = [-1.2, ..., 0.1]

It's raining cats and dogs = [0.8, 0.7]



Word-in-Context Vectors

[1.2, -0.3, ...]

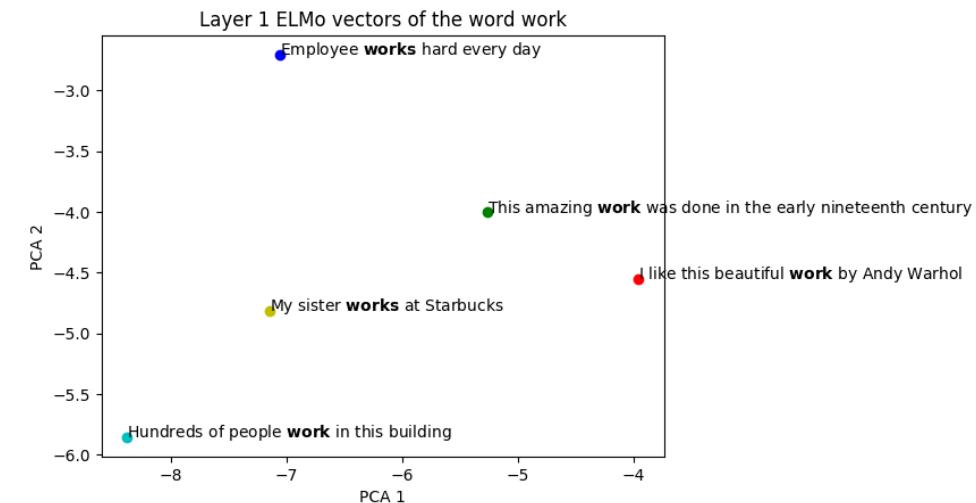


We have two Cats

[-0.5, 0.5, ...]



It's raining cats and dogs





03 TRANSFORMER

- Arsitektur Transformer
- Mekanisme Self-attention

Recall: Batasan Arsitektur RNN

- Sulit untuk memparalelkan komputasi secara efisien;
- Tidak dapat menangkap konteks kata pada kalimat secara utuh;
- Kinerja menurun jika urutan kalimat lebih panjang (vanishing gradient).



Attention!

High attention

Budi juara kelas dan dia senang. Sarah tidak juara kelas.

Low attention

Apa yang dimaksud **dia** atau **senang** dalam kalimat di atas? Ini merujuk ke **Budi**.

Ini cukup mudah dipahami manusia, tetapi tidak sesederhana jika dipahami oleh mesin. Pemahaman ini dapat diselesaikan menggunakan konsep **attention**.

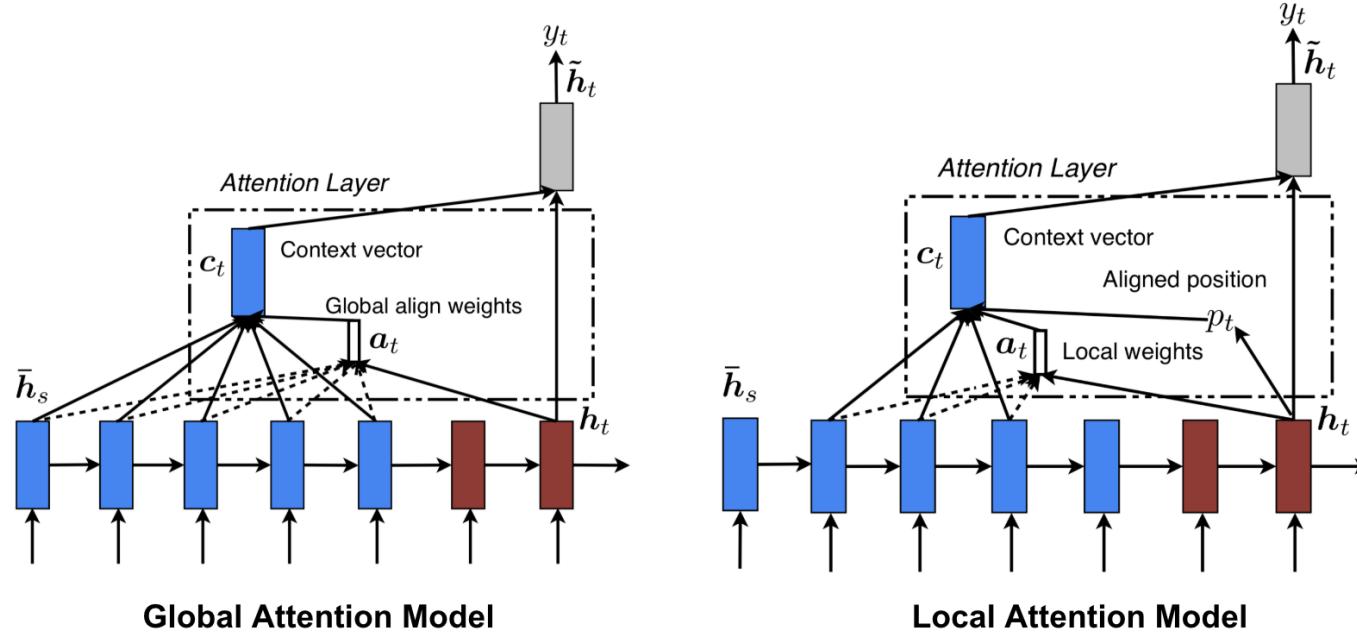
Baca selanjutnya:

A Family of Attention Mechanisms

<https://lilianweng.github.io/posts/2018-06-24-attention/#a-family-of-attention-mechanisms>

Global vs Local Attention

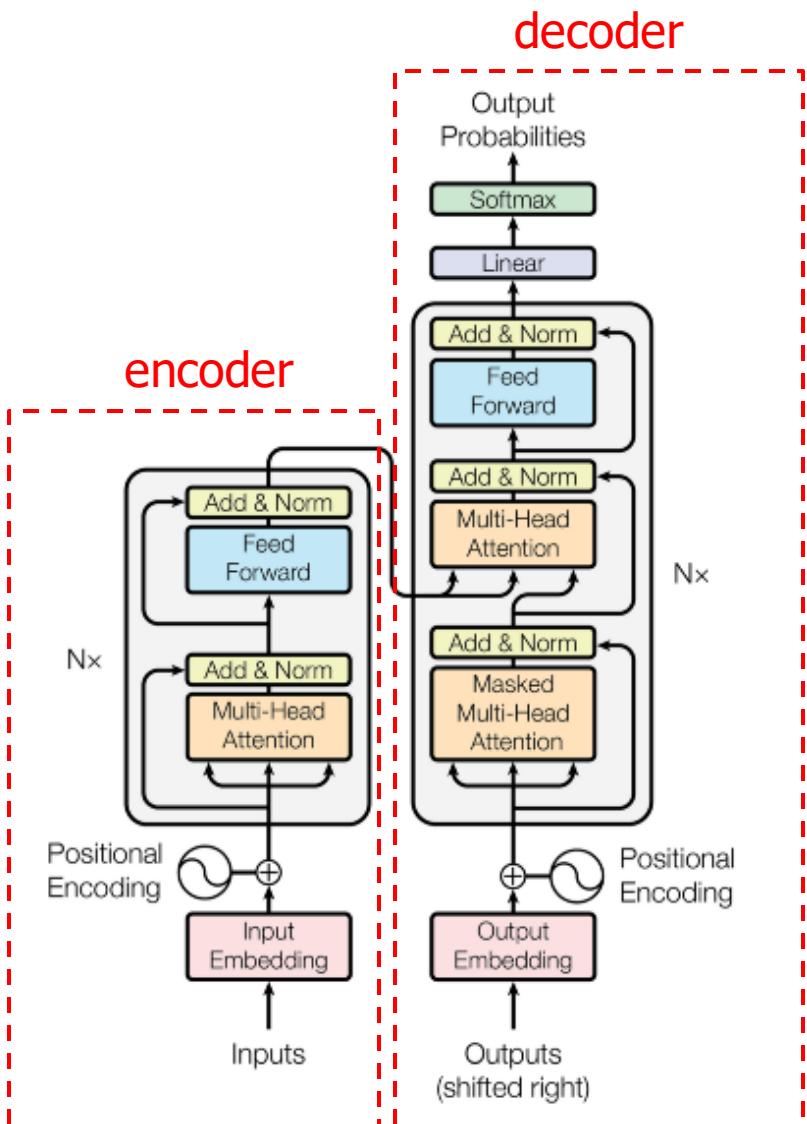
Attention dapat ditafsirkan secara luas sebagai vektor bobot satu kata dengan memperkirakan seberapa kuat kata itu berkorelasi dengan kata lain (bahkan dengan kata itu sendiri)



Baca selanjutnya:
M.-T. Luong, H. Pham, and C. D. Manning, "Effective Approaches to Attention-based Neural Machine Translation," arXiv.org, 2015, doi: 10.48550/arXiv.1508.04025.

<https://arxiv.org/abs/1508.04025>

Arsitektur Transformer



Model deep learning yang mengadopsi mekanisme attention. Mengganti seluruh lapisan RNN sehingga tidak harus memproses data secara berurutan.

Dirancang untuk machine translation (terdiri dari encoder & decoder).

Baca selanjutnya:

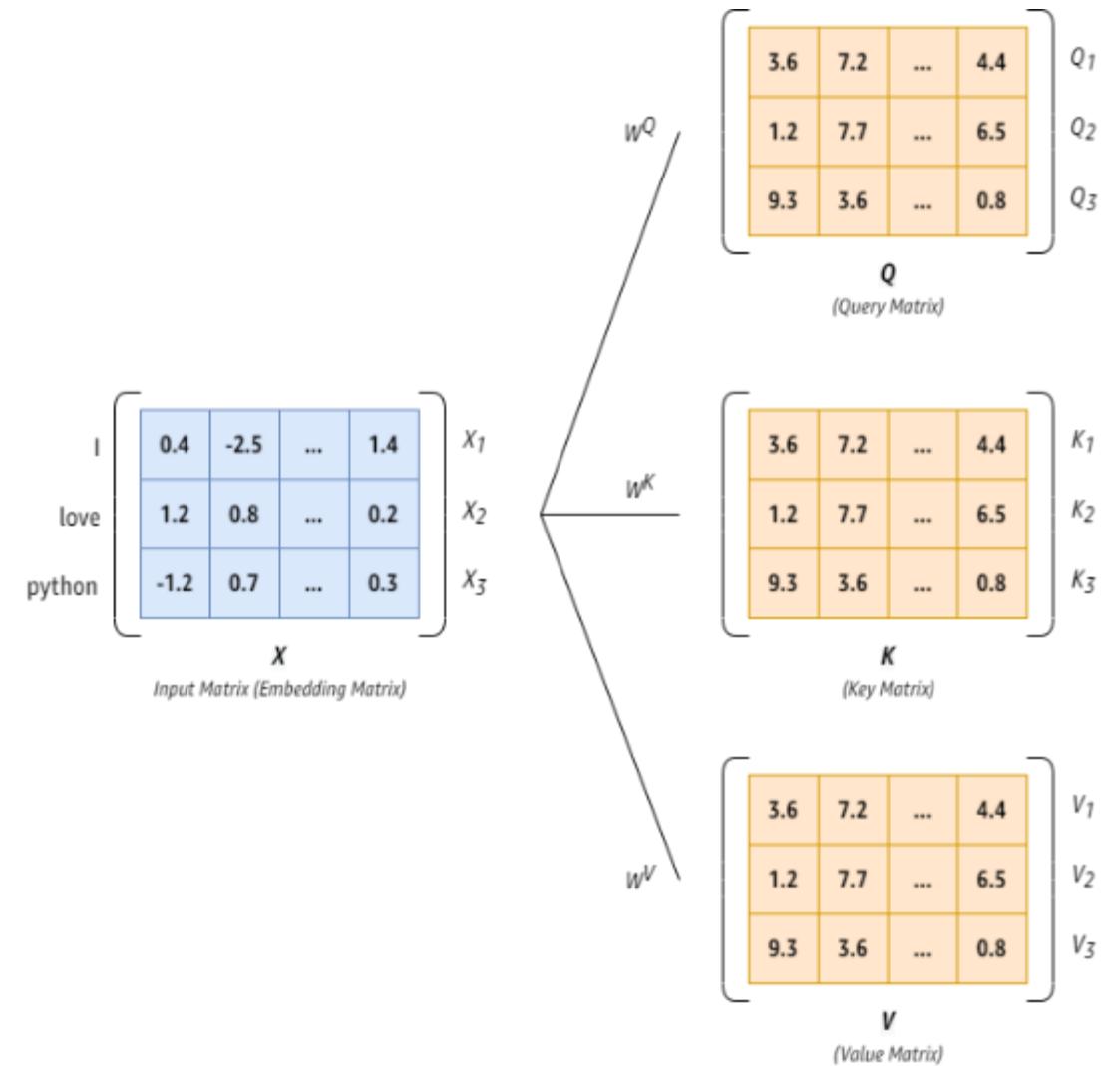
A. Vaswani et al., "Attention Is All You Need," arXiv.org, 2017, doi: 10.48550/arXiv.1706.03762.

<https://arxiv.org/abs/1706.03762>

Mekanisme Self-Attention (1)

$$\text{attention}(Q, K, V) = \text{softmax} \left(\frac{QK^T}{\sqrt{d_k}} \right) V$$

- Setiap baris input X adalah vektor embedding dari satu token.
- Dimensi matriks X adalah [panjang kalimat * dimensi embedding].
- Gunakan bobot W^Q , W^K , W^V untuk mengalikan matriks input X untuk mendapatkan matriks Q (query), K (key), V (value).
- Inisiasi bobot adalah random.



Mekanisme Self-Attention (2)

Hitung dot product antara matriks Q dan K . Mengapa?

- Setiap baris mewakili kesamaan antara satu token dan token lainnya.
- Misalnya, kita dapat memahami bahwa kata 'I' lebih terkait dengan dirinya sendiri daripada kata 'love' dan 'python'.

$$\begin{array}{c} \text{I} \\ \text{love} \\ \text{python} \end{array} \left[\begin{array}{cccc} 3.6 & 7.2 & \dots & 4.4 \\ 1.2 & 7.7 & \dots & 6.5 \\ 9.3 & 3.6 & \dots & 0.8 \end{array} \right]_{Q} \text{(Query Matrix)} \quad \bullet \quad \begin{array}{c} \text{I} & \text{love} & \text{python} \\ \begin{bmatrix} 1.2 & 0.6 & 3.6 \\ 8.1 & 7.7 & 0.2 \\ \vdots & \vdots & \vdots \\ 3.6 & 3.2 & 6.4 \end{bmatrix} \\ K_1 & K_2 & K_3 \end{array} = \begin{array}{c} \text{I} & \text{love} & \text{python} \\ \begin{bmatrix} 110 & 90 & 80 \\ 70 & 99 & 70 \\ 90 & 70 & 100 \end{bmatrix} \end{array}$$

Q^T

Mekanisme Self-Attention (3)

Bagi hasil dot product antara Q dan K dengan akar kuadrat dari **dimensi K** untuk mendapatkan gradien yang lebih stabil.

Lalu, lakukan normalisasi menggunakan fungsi **softmax**. Mengapa? Untuk menjadikan nilainya berkisar antara 0 hingga 1 dan jumlah skornya sama dengan 1.

$$\text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) = \begin{bmatrix} 110 & 90 & 80 \\ 70 & 99 & 70 \\ 90 & 70 & 100 \end{bmatrix} = \begin{bmatrix} 13.7 & 11.2 & 10 \\ 8.75 & 12.5 & 8.7 \\ 11.2 & 4.5 & 12.5 \end{bmatrix} = \begin{matrix} \text{I} & \text{love} & \text{python} \\ \text{love} & \begin{bmatrix} 0.90 & 0.07 & 0.03 \\ 0.02 & 0.80 & 0.02 \\ 0.21 & 0.03 & 0.76 \end{bmatrix} \\ \text{python} \end{matrix}$$

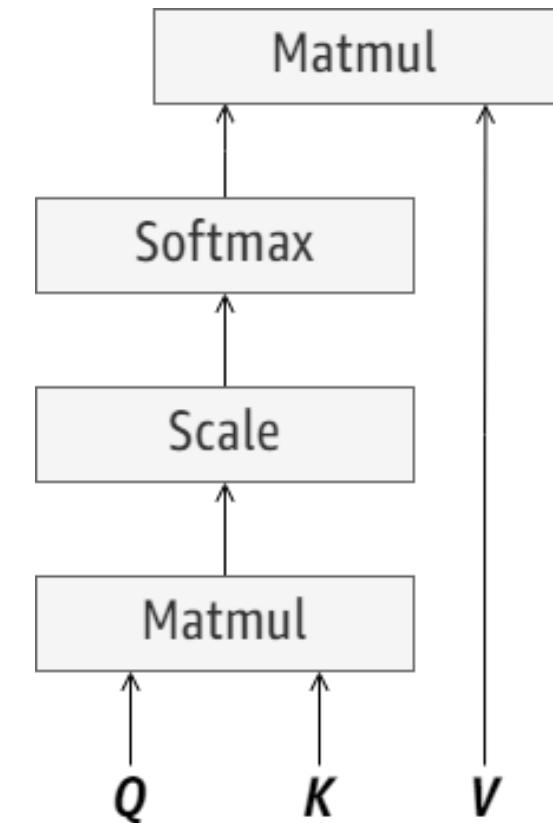
Mekanisme Self-Attention (4)

Langkah terakhir adalah menghitung matriks attention (Z) dengan mengalikan hasil langkah sebelumnya dengan V .

Ini menunjukkan bagaimana sebuah kata berhubungan dengan semua kata dalam suatu kalimat.

$$\begin{matrix} & \text{I} & \text{love} & \text{python} \\ \text{I} & \left[\begin{array}{ccc} 0.90 & 0.07 & 0.03 \\ 0.02 & 0.80 & 0.02 \\ 0.21 & 0.03 & 0.76 \end{array} \right] & \times & \left[\begin{array}{cccc} 3.6 & 7.2 & \dots & 4.4 \\ 1.2 & 7.7 & \dots & 6.5 \\ 9.3 & 3.6 & \dots & 0.8 \end{array} \right] \\ \text{love} & & & \\ \text{python} & & & \end{matrix} \quad V_1 = \begin{matrix} \text{I} \\ \text{love} \\ \text{python} \end{matrix} = \begin{matrix} 0.05 \\ 0.8 \\ 0.15 \end{matrix} Z_1$$

V
(Value Matrix)

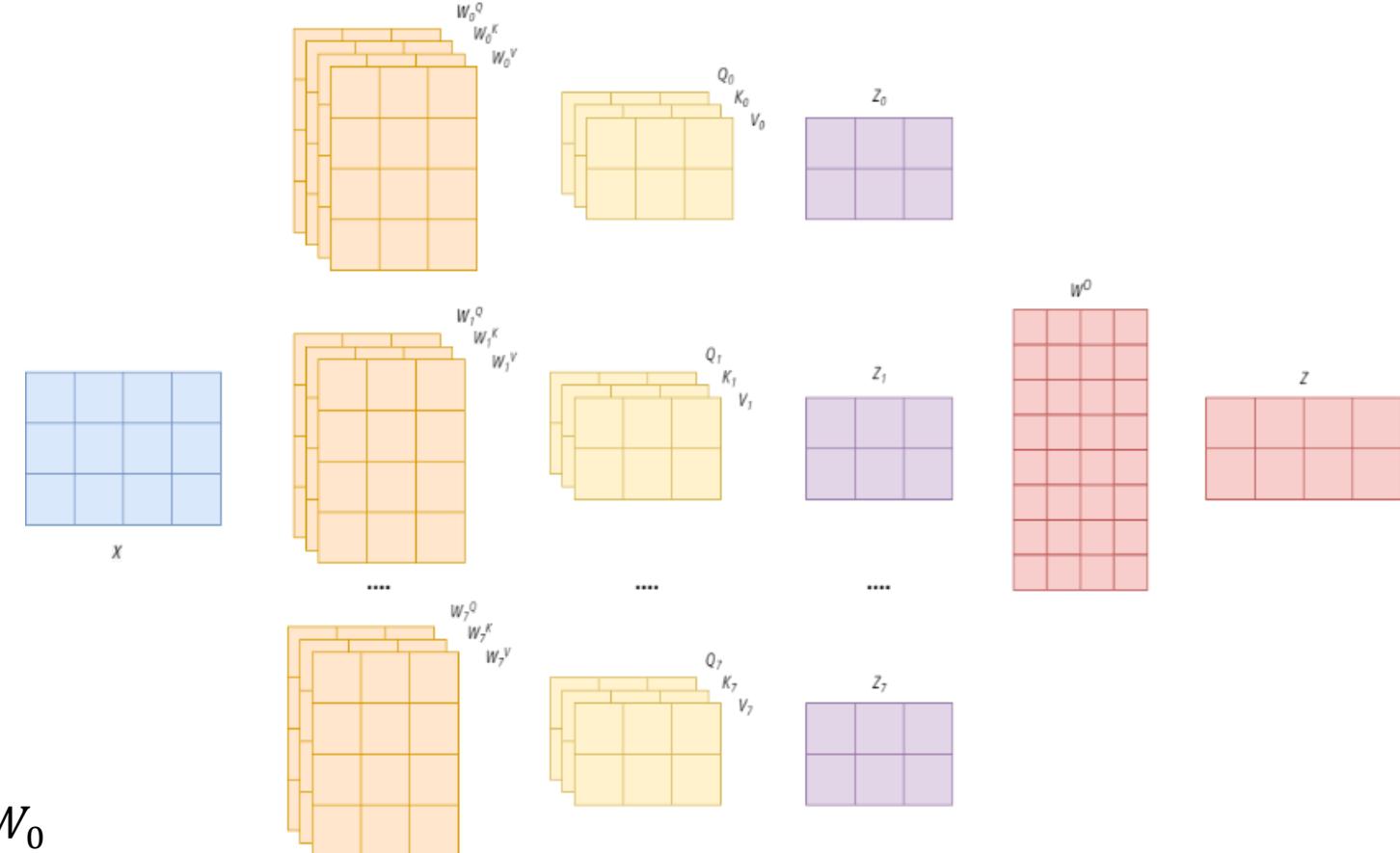


Mekanisme Multi-Head Attention

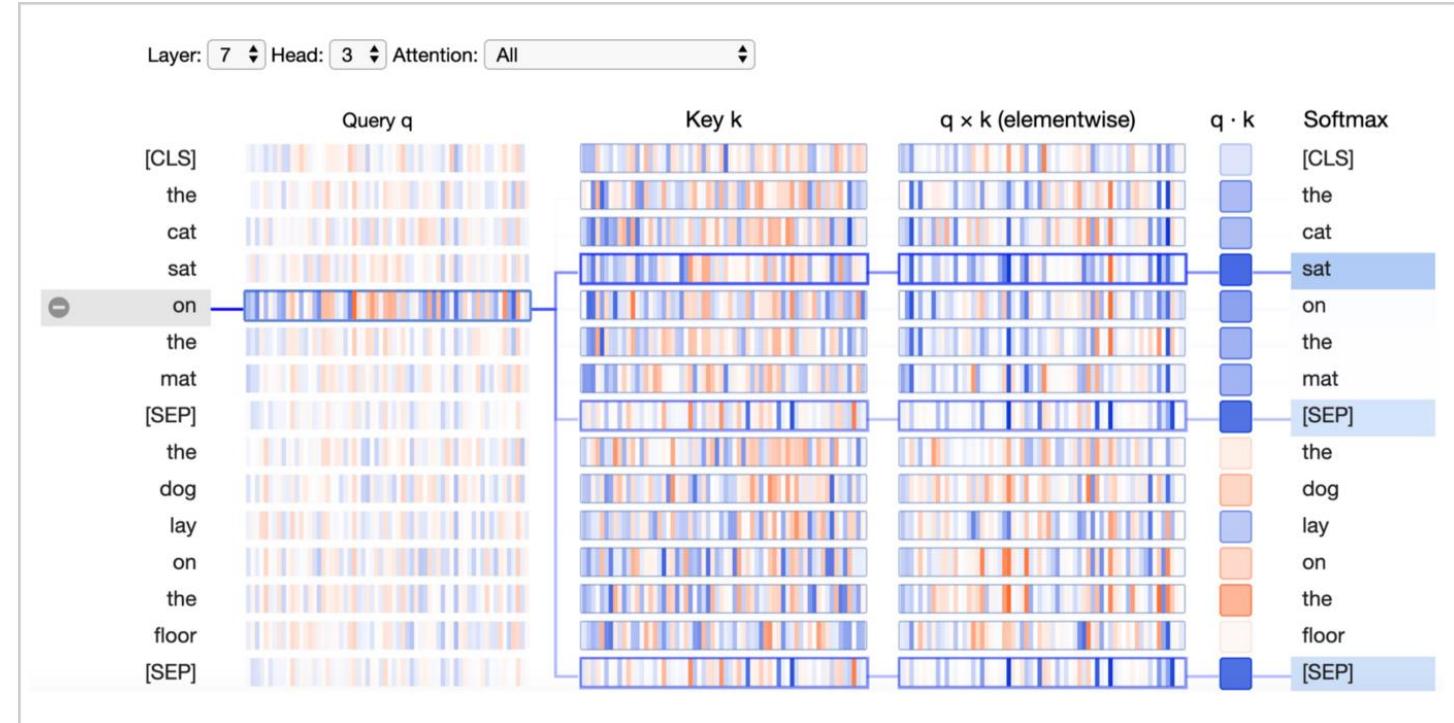
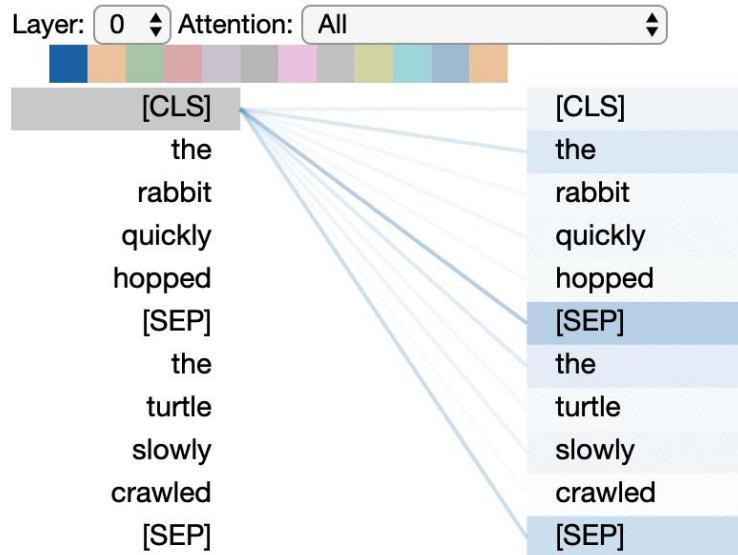
Mekanisme self-attention yang dihitung beberapa kali dalam arsitektur Transformer secara paralel dan independen.

Output dari independen self-attention kemudian digabungkan dan ditransformasikan secara linier ke dalam dimensi tertentu.

$$\text{MultiHead}(Q, K, V) = [\text{head}_1, \dots, \text{head}_h]W_0$$



Bagaimana hasilnya?



Baca selanjutnya:

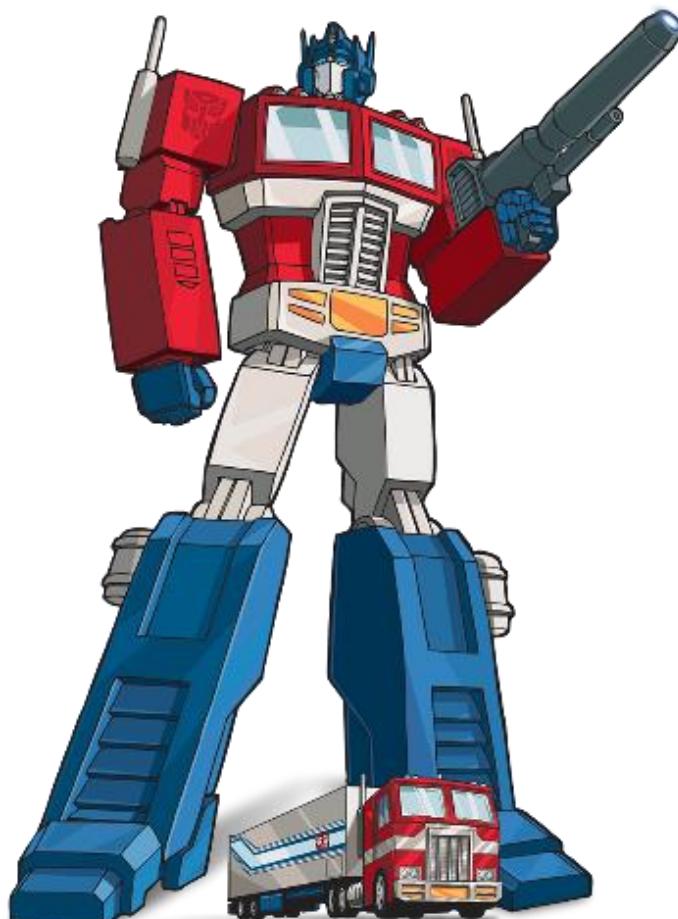
J. Vig, "A Multiscale Visualization of Attention in the Transformer Model," arXiv.org, 2019, doi: 10.48550/arXiv.1906.05714.

<https://arxiv.org/abs/1906.05714>

<https://github.com/jessevig/bertviz>

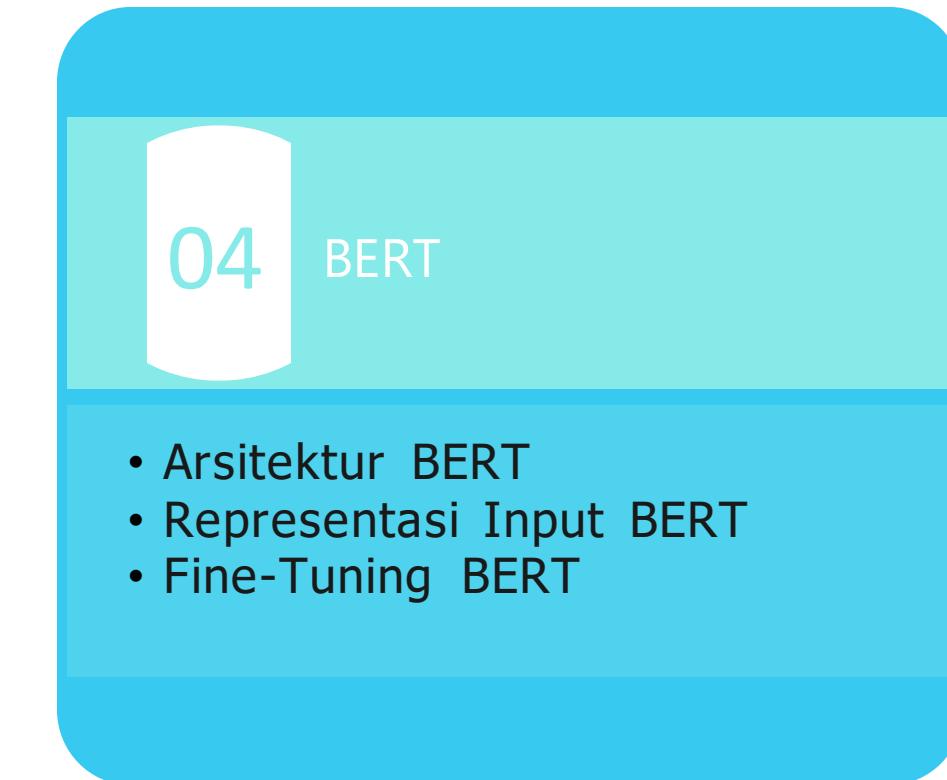
Model Berbasis Transformer

Model berbasis arsitektur Transformer adalah 'rahasia' di balik terobosan mutakhir pada bidang NLP.



Beberapa model berbasis Transformer yang populer dan mencapai performa tinggi:

- Bidirectional Encoder Representations from Transformers (**BERT**) oleh Google
- Generative Pre-trained Transformer (**GPT**) oleh OpenAI
- **XLNet** oleh Google

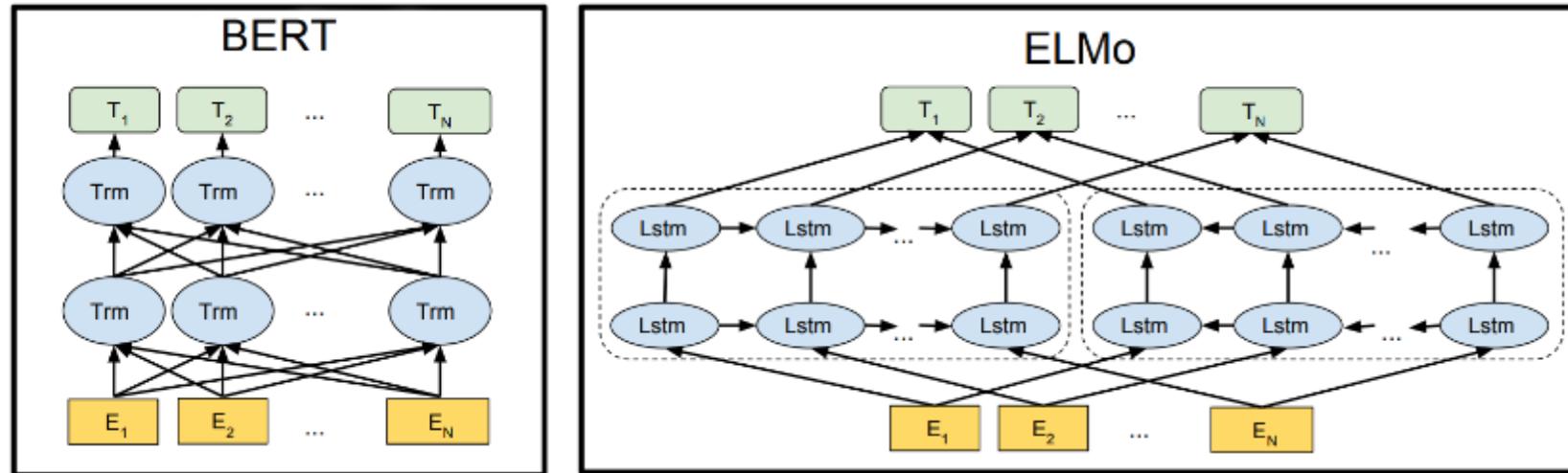


04 BERT

- Arsitektur BERT
- Representasi Input BERT
- Fine-Tuning BERT

BERT: Bidirectional Encoder Representations from Transformers

BERT merupakan arsitektur deep learning yang **truly bidirectional** sehingga mampu membaca konteks dari kiri-ke-kanan dan kanan-ke-kiri dengan dipelajari oleh jaringan yang sama.



Baca selanjutnya:

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," arXiv.org, 2018, doi: 10.48550/arXiv.1810.04805.

<https://arxiv.org/abs/1810.04805>

<https://github.com/google-research/bert>



Arsitektur BERT

BERT terdiri dari tumpukan **encoder** dari arsitektur **transformer** (disebut transformer block). Pada paper aslinya, disediakan 2 ukuran model:

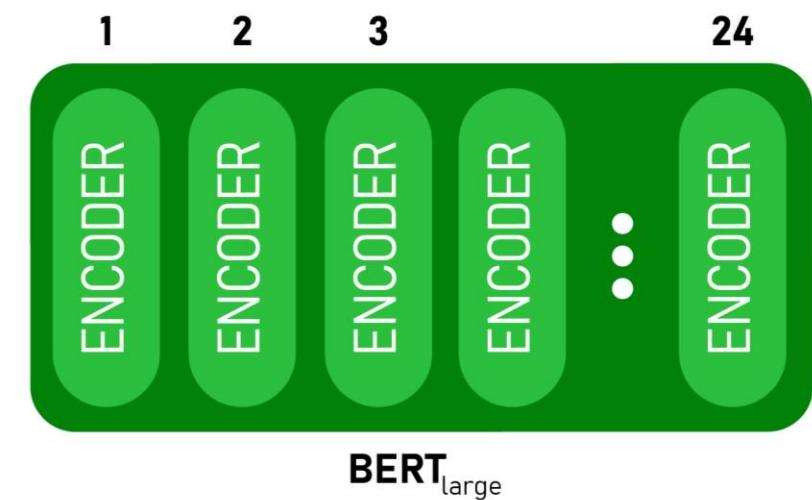
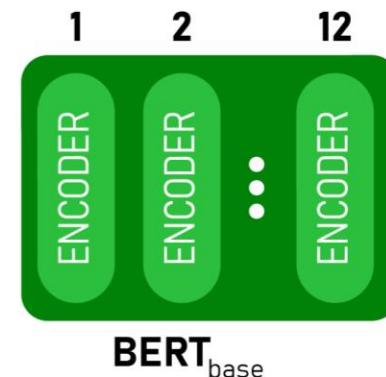
- $\text{BERT}_{\text{BASE}}$: $L=12, H=768, A=12$. Total parameter: 110 juta.
- $\text{BERT}_{\text{LARGE}}$: $L=24, H=1024, A=16$. Total parameter: 340 juta.

Dimana:

L = Jumlah layer atau encoder yang ditumpuk

H = Hidden size (ukuran vektor Q , K , dan V)

A = Jumlah attention heads



Arsitektur BERT (Smaller)

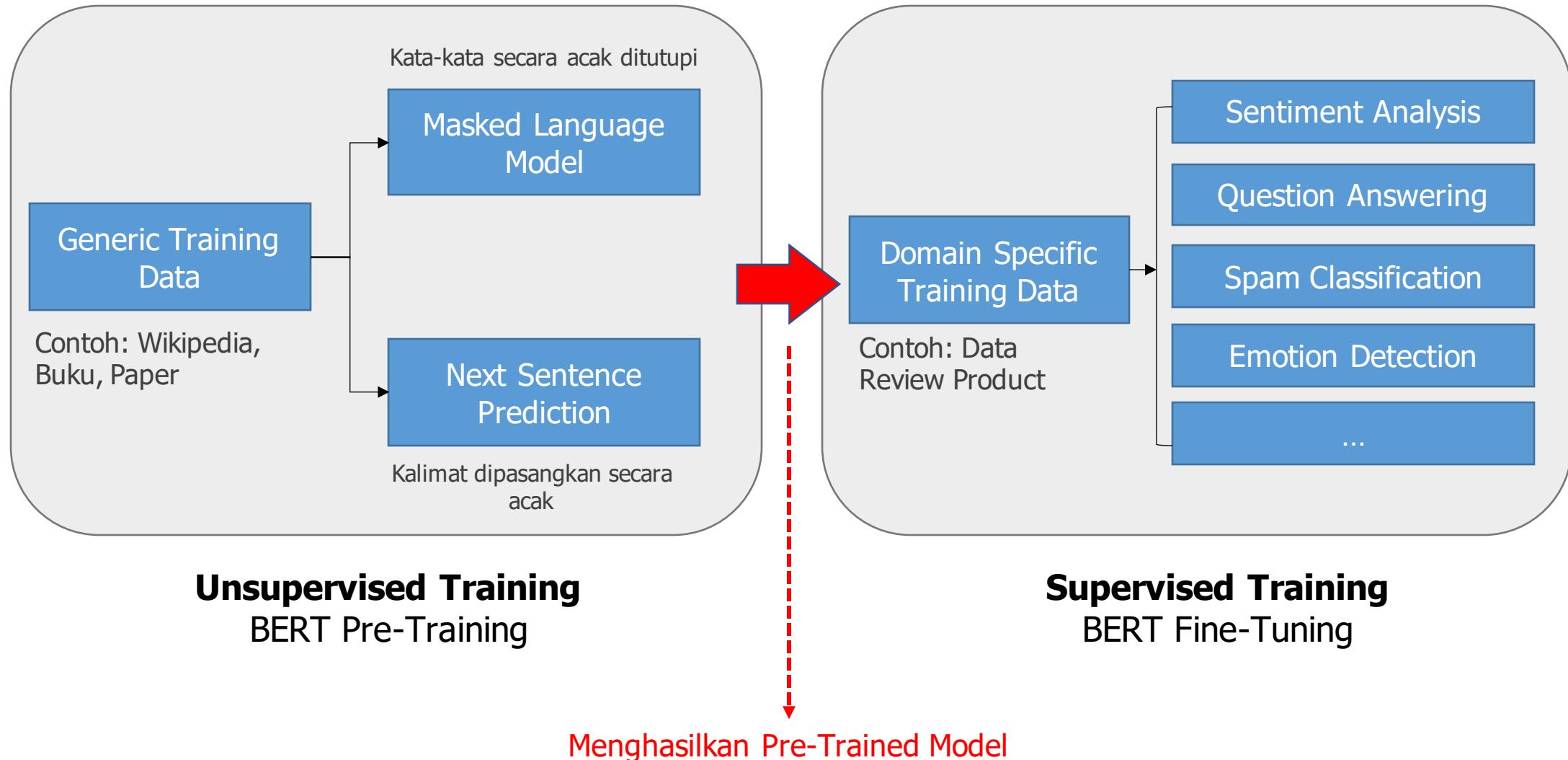
| | H=128 | H=256 | H=512 | H=768 |
|------|-------------------------------|-------------------------------|---------------------------------|--------------------------------|
| L=2 | 2/128 | 2/256 | 2/512 | 2/768 |
| L=4 | 4/128 (BERT _{TINY}) | 4/256 (BERT _{MINI}) | 4/512 (BERT _{SMALL}) | 4/768 |
| L=6 | 6/128 | 6/256 | 6/512 | 6/768 |
| L=8 | 8/128 | 8/256 | 8/512 (BERT _{MEDIUM}) | 8/768 |
| L=10 | 10/128 | 10/256 | 10/512 | 10/768 |
| L=12 | 12/128 | 12/256 | 12/512 | 12/768 (BERT _{BASE}) |

Baca selanjutnya:

I. Turc, M.-W. Chang, K. Lee, and K. Toutanova, "Well-Read Students Learn Better: On the Importance of Pre-training Compact Models," arXiv.org, 2019, doi: 10.48550/arXiv.1908.08962.

<https://arxiv.org/abs/1908.08962>

BERT Part



BERT Pre-Trained Model



<https://huggingface.co/models?search=bert>

Indonesian BERT Pre-Trained Model

| # | Model Name | Dataset | Source Paper |
|---|-------------------|---|---|
| 1 | BERT Multilingual | 104 Bahasa dari Wikipedia Dump | J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," 2018, doi: 10.48550/arXiv.1810.04805 |
| 2 | IndoBERT | Indo4B (4 miliar kata / 250 juta kalimat) | B. Wilie et al., "IndoNLU: Benchmark and Resources for Evaluating Indonesian Natural Language Understanding," 2020, doi: 10.48550/arXiv.2009.05387 . |
| 3 | IndoBERT | <ul style="list-style-type: none">- Indonesian Wikipedia (74 juta kata)- Artikel berita (55 juta kata)- Web corpus (90 juta kata) | F. Koto, A. Rahimi, J. H. Lau, and T. Baldwin, "IndoLEM and IndoBERT: A Benchmark Dataset and Pre-trained Language Model for Indonesian NLP," 2020, doi: 10.48550/arXiv.2011.00677 . |
| 4 | IndoBERTweet | 26 juta tweets (409 juta token kata) | F. Koto, J. H. Lau, and T. Baldwin, "IndoBERTweet: A Pretrained Language Model for Indonesian Twitter with Effective Domain-Specific Vocabulary Initialization," 2021, doi: 10.48550/arXiv.2109.04607 . |

BERT PreTraining: Masked Language Model

Tutupi (masking) $k\%$ dari kata-kata input secara acak, lalu model akan memprediksi kata-kata yang ditutupi tersebut.

Input: the man went to the [MASK1] he bought a [MASK2] of milk
Labels: [MASK1] = store; [MASK2] = gallon

BERT PreTraining: Next Sentence Prediction

Digunakan untuk mempelajari hubungan antar kalimat. Prediksi apakah Kalimat B adalah kalimat aktual yang menghasilkan Kalimat A, atau kalimat acak.

Sentence A: the man went to the store.

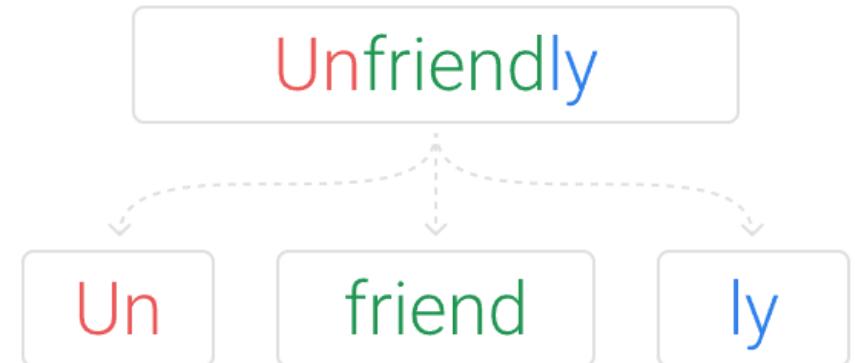
Sentence B: he bought a gallon of milk.

Label: IsNextSentence

BERT Tokenizer

- BERT menggunakan **WordPiece tokenizer** untuk menghasilkan dictionary. Ukuran dictionary yang digunakan adalah 30.000 token.
- Kata asli **dipecah** menjadi **sub-kata** dan **karakter** yang lebih kecil.
- Kata-kata yang tidak ada pada dictionary direpresentasikan sebagai subkata dan karakter (out-of-vocabulary problem).

```
[ 'em' , '##bed' , '##ding' , '##s' ]
```



Baca selanjutnya:

Y. Wu et al., "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation," arXiv.org, 2016, doi: 10.48550/arXiv.1609.08144.

<https://arxiv.org/abs/1609.08144>

BERT Cased vs Uncased

Cased → Teks tidak diubah sama sekali.

Uncased → Teks diubah menjadi lowercase sebelum langkah tokenisasi menggunakan WordPiece. Selain itu, teks dengan penanda aksen akan dihapus, seperti Opèn akan diubah menjadi open.

***BERT uncased lebih baik daripada BERT cased** di sebagian besar tugas kecuali pada tugas di mana informasi teks lebih penting (seperti, Named Entity Recognition and Part-of-Speech tagging).

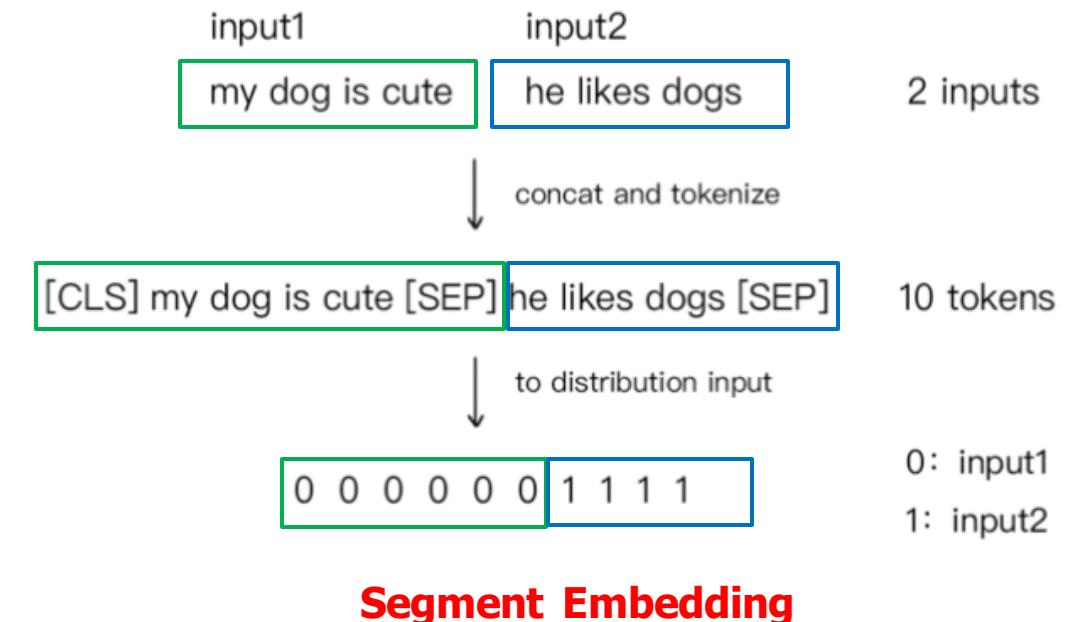
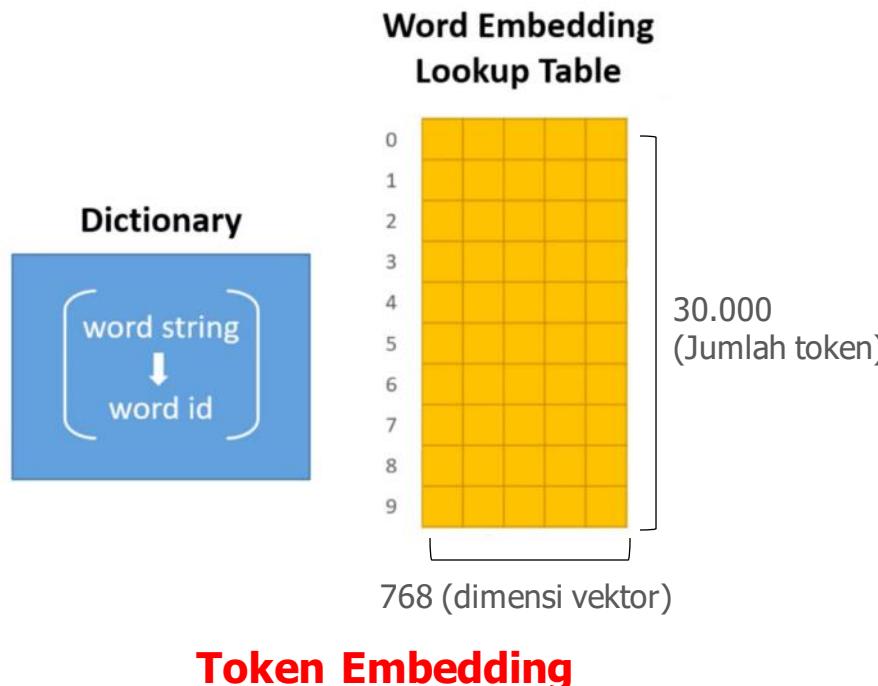
BERT Special Tokens

BERT **mewajibkan** penggunaan token khusus untuk dapat memahami input dengan benar, antara lain:

- Token **[CLS]** ditambahkan di awal kalimat yang digunakan untuk tugas klasifikasi. Pada BERT, token **[CLS]** menunjukkan hidden state terakhir dari seluruh kalimat.
- Token **[SEP]** ditambahkan di akhir kalimat yang digunakan untuk memisahkan kalimat seperti pada tugas “Next Sentence Prediction”.
- Token **[PAD]** ditambahkan untuk mengisi token kosong karena BERT menerima panjang kalimat yang tetap sebagai input.
- Token **[MASK]** digunakan untuk tugas “Masked Language Model” untuk menutupi sebagian kata.
- Token **[UNK]** digunakan untuk mengidentifikasi kata-kata yang tidak dikenal, bahkan setelah proses tokenisasi.

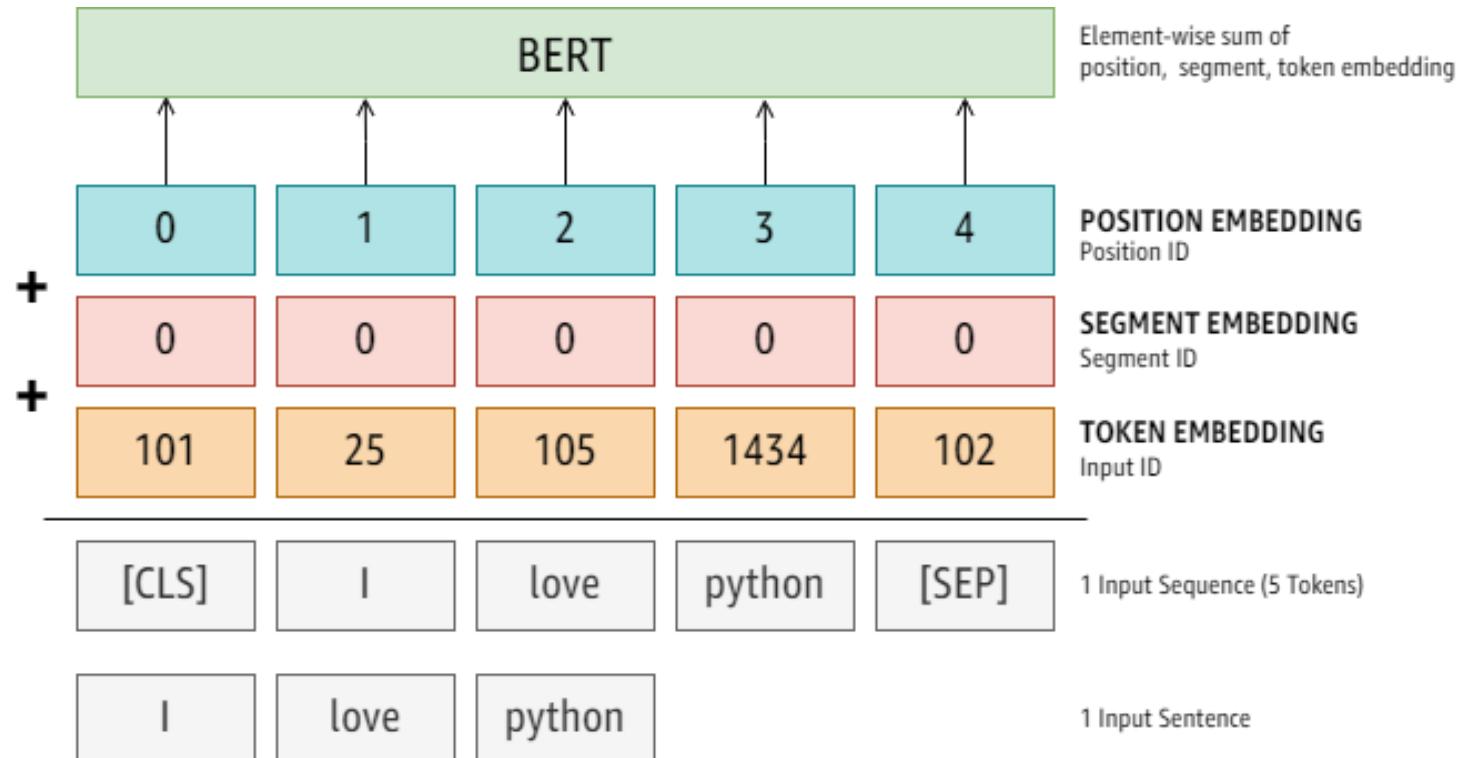
Representasi Input BERT (1)

- **Token embedding** menunjukkan id token pada dictionary yang dihasilkan pada proses Tokenizer.
- **Segment embedding** menunjukkan urutan kalimat. BERT harus mengetahui apakah suatu token termasuk dalam kalimat A atau kalimat B.
- **Position embedding** menunjukkan posisi kata dalam kalimat.



Representasi Input BERT (2)

- **Kalimat** dapat berupa rentang **teks** (kata).
- **Urutan** (sequences) mengacu pada **urutan token input**. Seluruh input ke BERT harus diberikan dalam satu urutan.
- Setiap token adalah jumlah dari tiga embeddings yang berbeda.



BERT Fine Tuning

Fine-tuning adalah proses **melatih model** menggunakan **model yang sudah dilatih** sebelumnya untuk tugas yang berbeda. Sangat erat kaitannya dengan transfer learning.

Mengapa?

- Membutuhkan data latih (domain specific) yang sedikit;
- Waktu dan sumber daya komputasi lebih cepat daripada melatih model dari awal;
- Performa model jauh lebih baik.

BERT Downstream Task

Merujuk pada **supervised-learning task** yang memanfaatkan **pre-trained model**.

Downstream task pada area NLP meliputi:

- Sequence (text) classification;
- Question answering;
- Text generation;
- Named entity recognition (NER);
- Summarization;
- Machine translation.

*BERT **tidak bisa** digunakan untuk text generation dan machine translation.

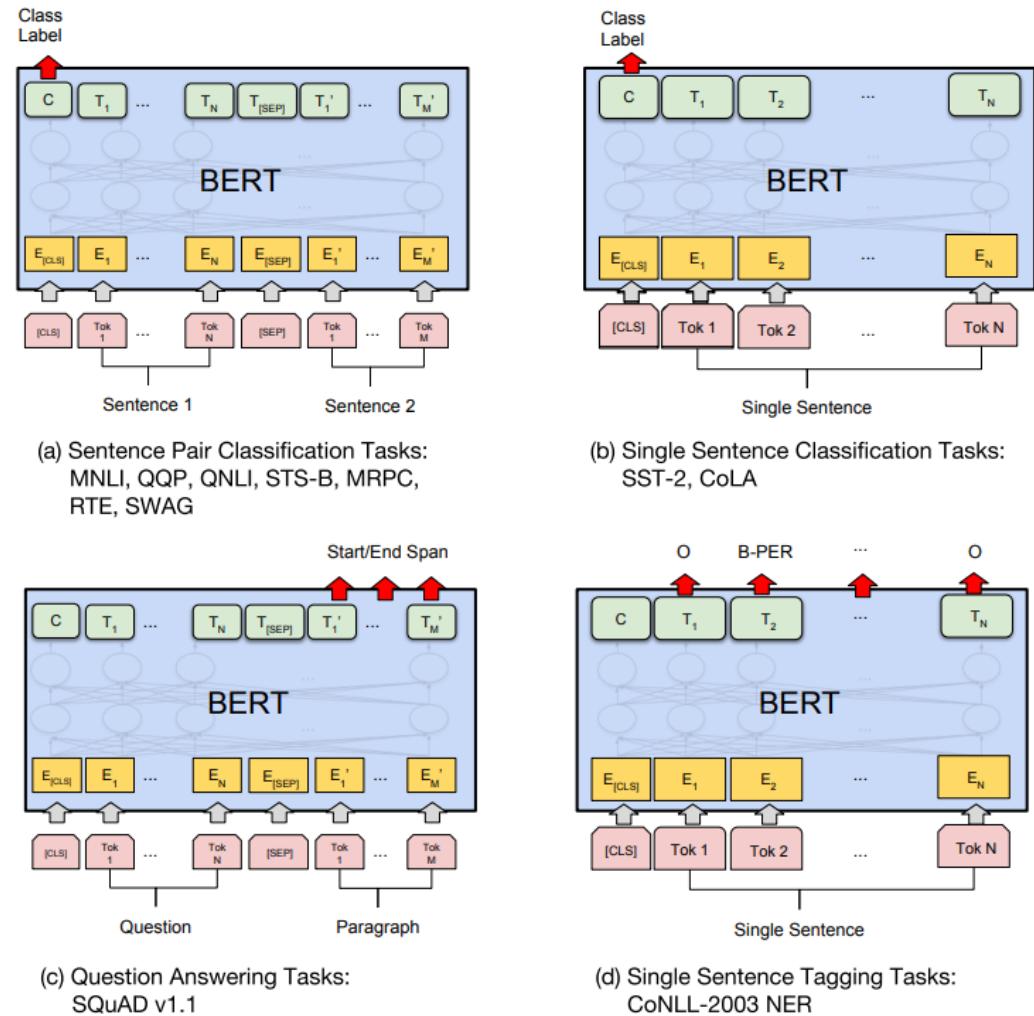


Figure 4: Illustrations of Fine-tuning BERT on Different Tasks.

BERT Fine Tuning Procedure

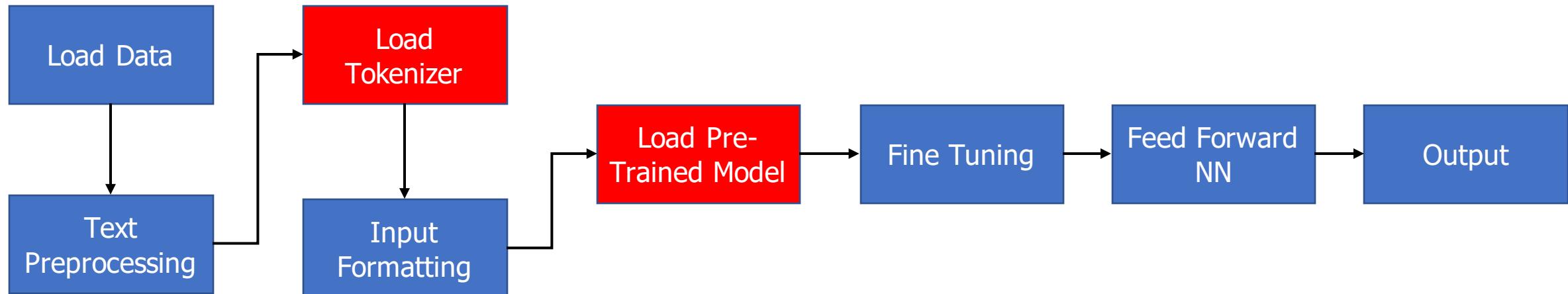
Pada makalah aslinya, penulis menyarankan nilai hyperparameter untuk fine tuning:

- **Batch size:** 16, 32
- **Learning rate (Adam):** 5e-5, 3e-5, 2e-5
- **Number of epochs:** 2, 3, 4

***Nilai** hyperparameter yang **optimal** adalah **tergantung tugas** yang diselesaikan.

BERT Fine Tuning Scheme

Secara umum, skema fine-tuning BERT untuk tugas klasifikasi dapat dilakukan mengikuti bagan berikut:



- *Tokenizer yang digunakan harus sama dengan pre-trained model yang digunakan.
- **Feed Forward Neural Network digunakan untuk pengklasifikasi.

Tools

Library yang dapat digunakan untuk implementasi arsitektur berbasis Transformer (termasuk BERT):

- Transformers (<https://huggingface.co>)
- Simple Transformers (<https://simpletransformers.ai>)
- Fast-Bert (<https://github.com/utterworks/fast-bert>)

Implementasi dapat menggunakan dua framework deep learning: TensorFlow maupun PyTorch.

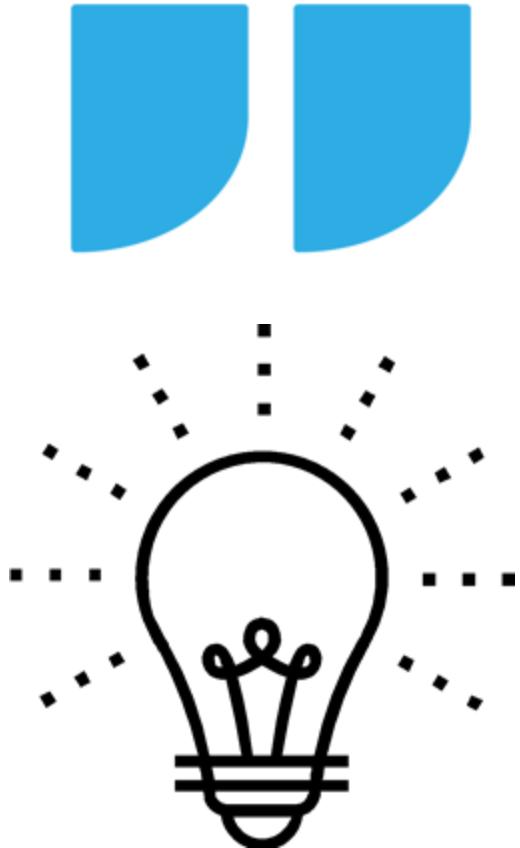


05 KESIMPULAN

- Ringkasan
- Kuis

Ringkasan

1. Transfer learning mengacu pada melatih model kemudian menggunakan model tersebut untuk menyelesaikan tugas lainnya.
2. Word2Vec gagal menangkap informasi kontekstual yang terkandung pada seluruh kalimat (context free manner).
3. Attention mengacu pada pembobotan kata dengan melihat korelasi kata tersebut pada seluruh kalimat (bahkan kata itu sendiri).
4. Transformer merupakan model deep learning yang menggunakan mekanisme attention untuk mengganti seluruh lapisan berbasis RNN. Transformer terdiri dari encoder dan decoder (dirancang untuk machine translation)
5. BERT merupakan model deep learning yang menggunakan tumpukan encoder dari arsitektur Transformer.
6. Fine-tuning adalah proses melatih model menggunakan model yang sudah dilatih sebelumnya. Cara menerapkan konsep transfer learning.\
7. Fine-tuning BERT membutuhkan input formatting dengan menambahkan special token.



Kuis

Pertanyaan

BERT terdiri dari tumpukan arsitektur Transformer

- A. Decoder
- B. Encoder
- C. Encoder - Decoder
- D. Self Attention



Kuis

Pertanyaan

BERT terdiri dari tumpukan arsitektur Transformer

- A. Decoder
- B. Encoder**
- C. Encoder - Decoder
- D. Self Attention



Jawaban: B



TERIMA KASIH

Orbit Future Academy

PT Orbit Ventura Indonesia
Center of Excellence (Jakarta Selatan)
Gedung Veteran RI, Lt.15
Unit Z15-002, Plaza Semanggi
Jl. Jenderal Sudirman Kav.50, Jakarta
12930, Indonesia

- Jakarta Selatan/Pusat
- Jakarta Barat/BSD
- Kota Bandung
- Kab. Bandung
- Jawa Barat

Hubungi Kami

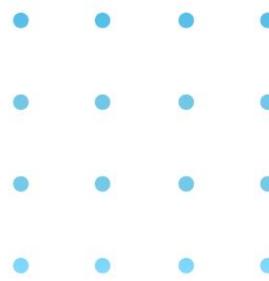
Director of Sales & Partnership
ira@orbitventura.com
+62 858-9187-7388

Social Media

-  [Orbit Future Academy](#)
-  [@OrbitFutureAcademyIn1](#)
-  [OrbitFutureAcademy](#)
-  [Orbit Future Academy](#)



AI Mastery Course



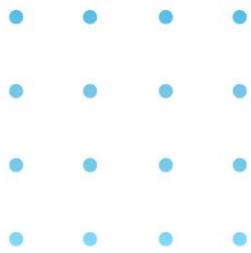
Module 10

Natural Language Processing (NLP)

Section 9

Speech Recognition

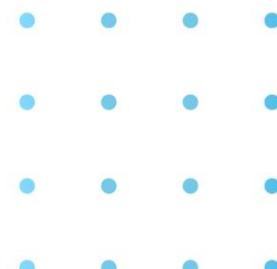




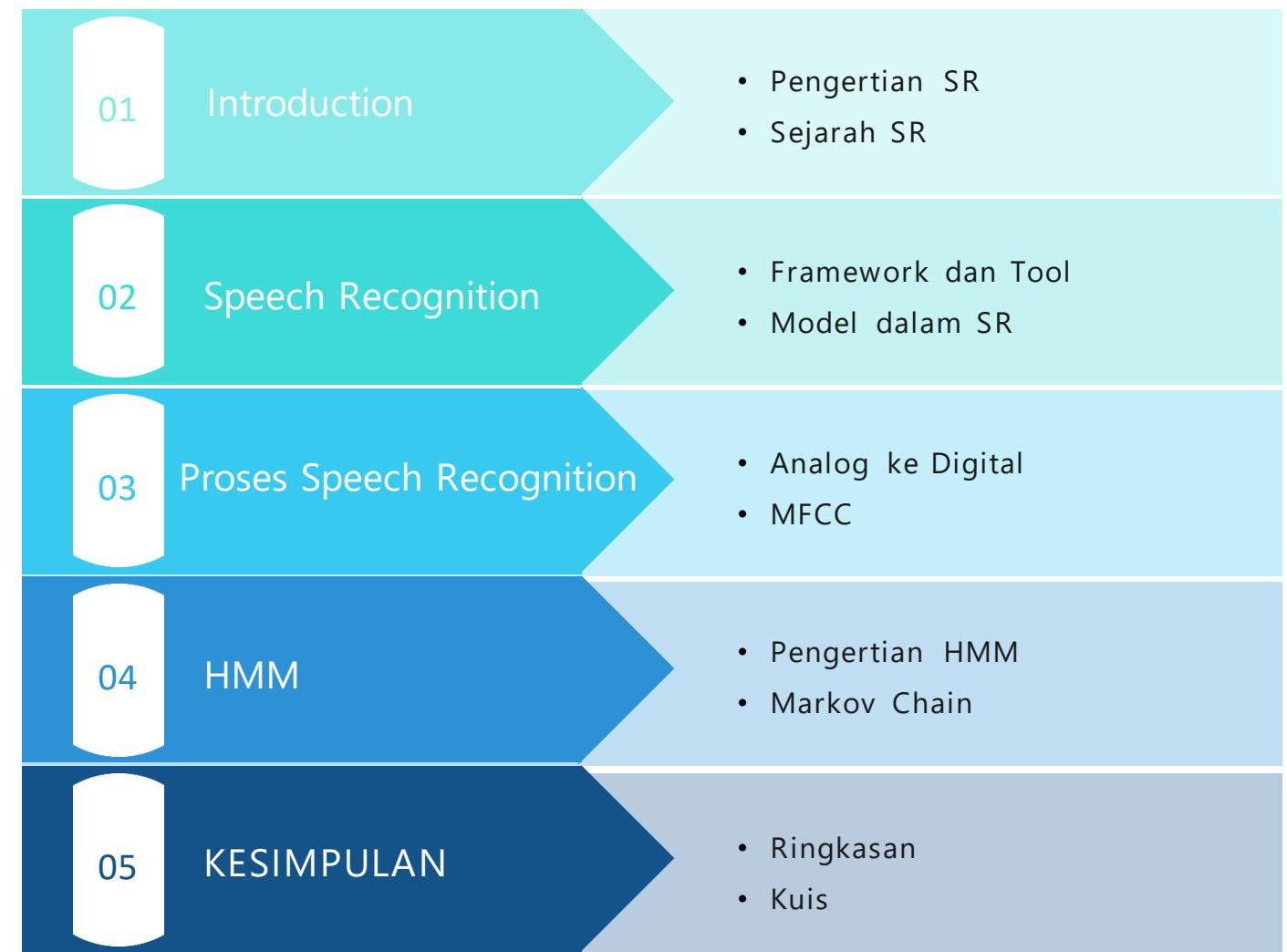
Learning Objectives

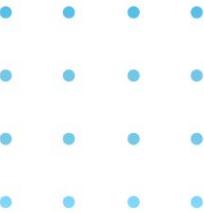
Di akhir modul ini, Anda akan mendapatkan:

- Memahami pengertian Speech Recognition
- Proses mengubah sinyal analog ke digital
- Ekstraksi Fitur dengan MFCC
- Flow Speech Recognition
- Hidden Markov Model



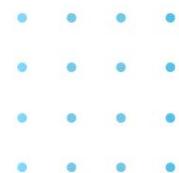
Agenda





01 INTRODUCTION

- Pengertian SR
- Sejarah SR



Speech Recognition



Speech Recognition

- Speech Recognition atau yang biasa dikenal dengan ***Automatic Speech Recognition (ASR)*** merupakan suatu pengembangan teknik dan sistem yang memungkinkan komputer untuk **menerima masukan berupa kata yang diucapkan**.
- Teknologi ini memungkinkan suatu perangkat untuk mengenali dan memahami kata-kata yang diucapkan dengan cara mengubah sinyal analog berupa gelombang suara menjadi **sinyal digital** dan mencocokkan sinyal digital tersebut dengan suatu pola tertentu yang tersimpan dalam suatu perangkat.

Perbedaan **Speech Recognition** **Voice Recognition**

Speech Recognition vs Voice Recognition

- **Pengenalan ucapan (Speech Recognition)** digunakan untuk mengidentifikasi kata-kata dalam bahasa lisan atau percakapan lisan.
- **Pengenalan suara (Voice Recognition) adalah** teknologi untuk mengidentifikasi suara individu.

Sejarah Speech Recognition



1784

Wolfgang von Kempelen creates the **Acoustic-Mechanical Speech Machine** in Vienna

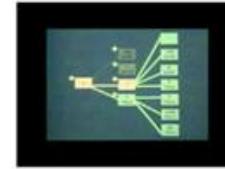


Thomas Edison invents the first dictation machine



1952

IBM Shoobox can understand 16 English words



1971

IBM Tangora, using the Hidden Markov Model, predicts upcoming phonemes in speech



2006

The National Security Agency (NSA) starts using speech recognition to isolate key words in recorded speech



2008



Google launches a voice search app, bringing speech recognition to mobile devices

2011



Apple announces Siri, ushering in the age of the voice-enabled digital assistant

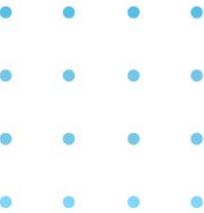


1962

Harpy, created at Carnegie Mellon University, can comprehend 1,011 words - and some phrases



1986



02 Speech Recognition

- Framework dan Tool
- Model dalam SR



Frameworks

Beberapa **frameworks** open-source untuk Automatic Speech Recognition

- CMU Sphinx - <https://cmusphinx.github.io/>
- Kaldi - <https://github.com/kaldi-asr/kaldi>
- ESPnet - <https://github.com/espnet/espnet>

Audio Processing

Dua **toolkit** audio processing yang open-source untuk Automatic Speech Recognition

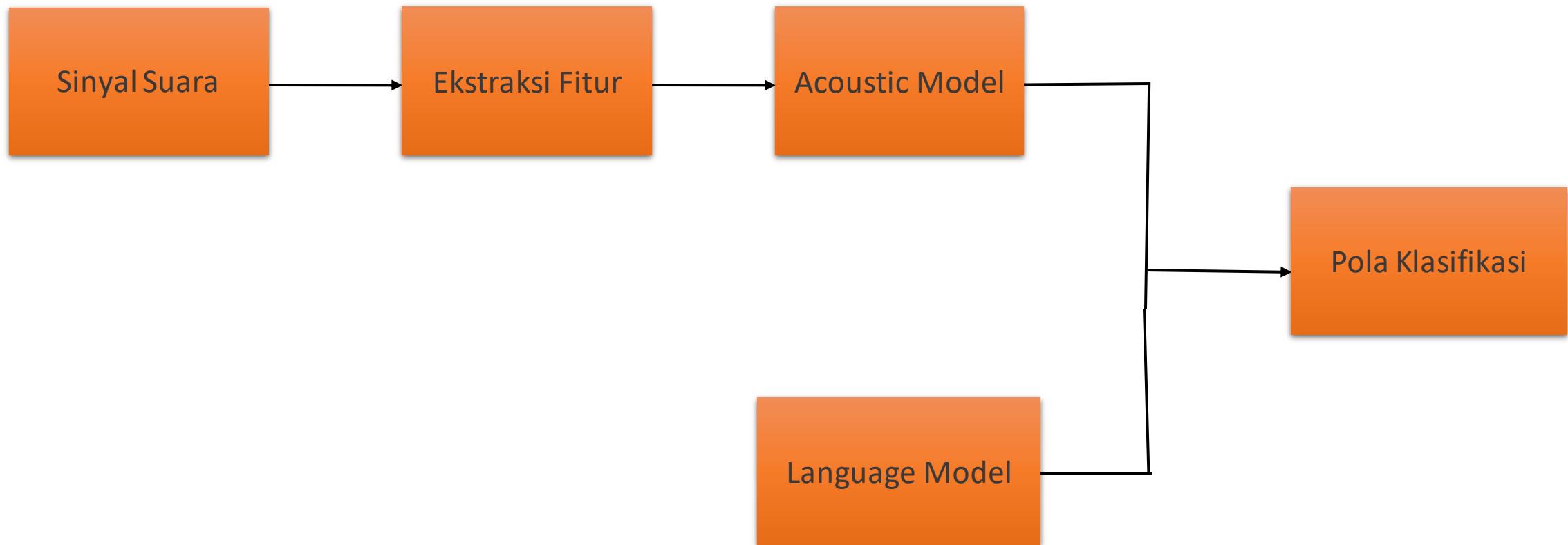
SoX – Sound eXchange

Toolkit dan library yang dapat digunakan untuk memanipulasi audio. Tools ini mengimplementasikan banyak format file dan dapat digunakan untuk memutar, mengonversi, dan memanipulasi file audio.

Librosa

Library Python yang dapat digunakan untuk analisis audio, mengekstraksi fitur, dan pemrosesan sinyal digital (Digital Signal Processing, DSP).

Speech Recognition Flow



Model dalam Speech Recognition

Model Akustik

Ini mewakili hubungan antara unit linguistik ucapan (phonem) dan sinyal audio.

Model Bahasa

Pada model ini suara dicocokkan dengan urutan kata untuk membedakan antara kata-kata yang mirip.

Poin Penting pada SR

Algoritma pada Speech Recognition harus **mampu memproses dan mengubah audio menjadi teks**. Adapun poin penting yang perlu diperhatikan adalah sebagai berikut :

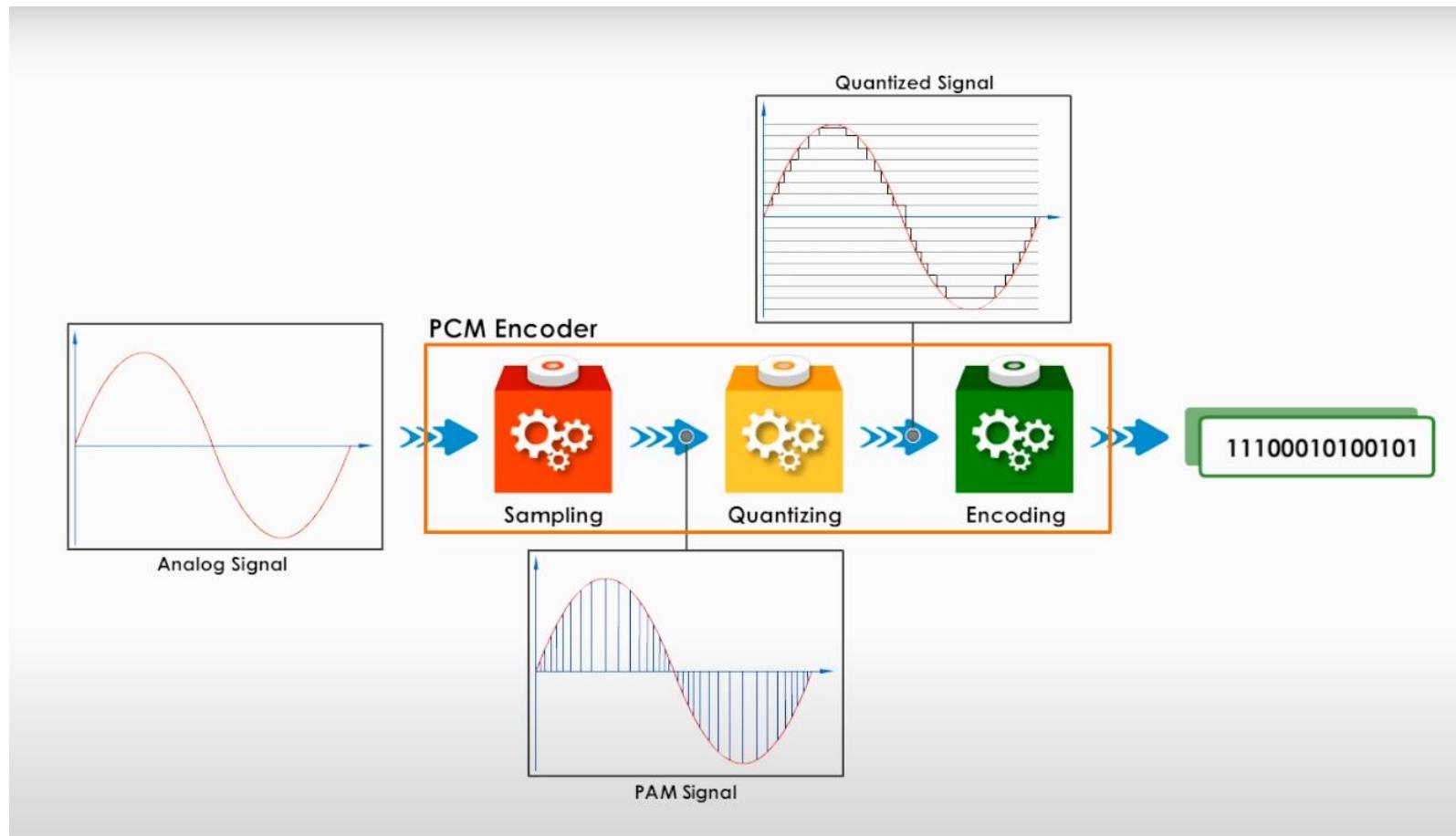
- Gaya bicara
- Bahasa
- Dialek & Aksen
- Perbedaan frasa



03 Proses Speech Recognition

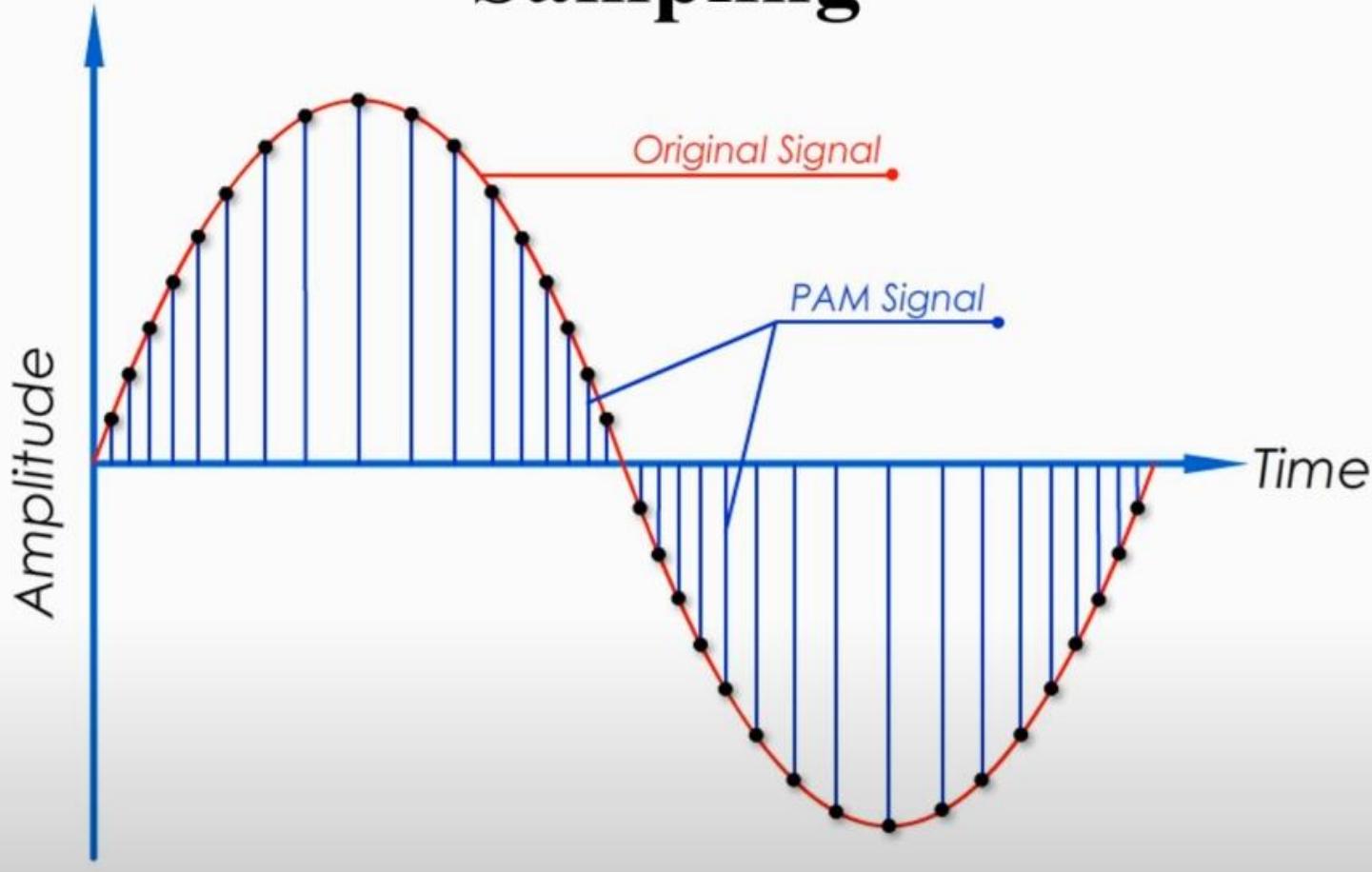
- Analog ke Digital
- MFCC

Sinyal Analog ke Digital



Sumber : <https://youtu.be/HIGJ6xxbz8s>

Sampling



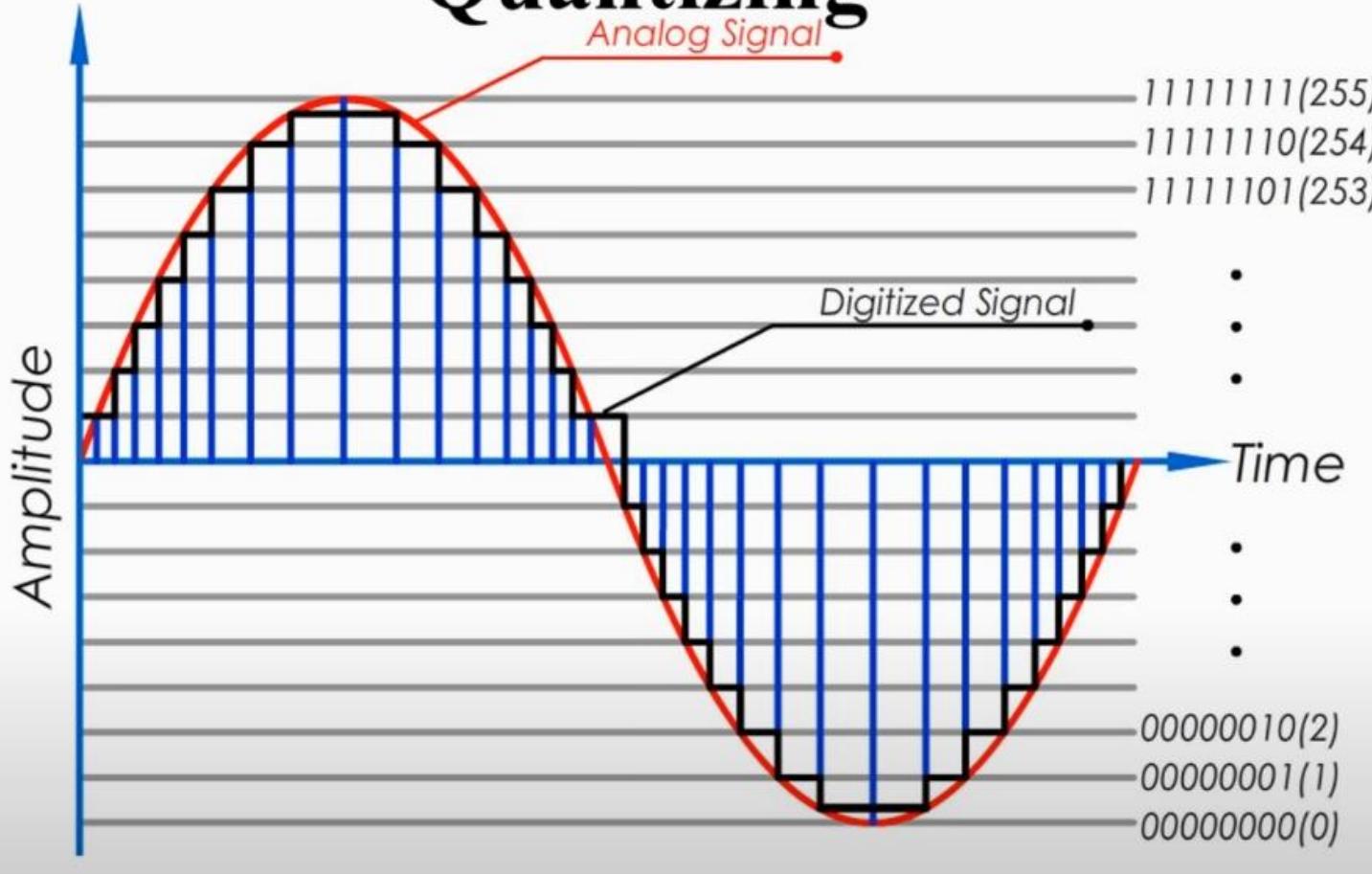
Pulse Amplitude Modulation (PAM) yaitu sample dengan nilai diskrit pada amplitudo terhadap waktu

Sample rate – Jumlah frame persecond (Hz)
Contoh standar sample rate:

- Telepon 8 KHz
- MP3 44 KHz
- Bluray 1 MHz

Quantizing

Analog Signal



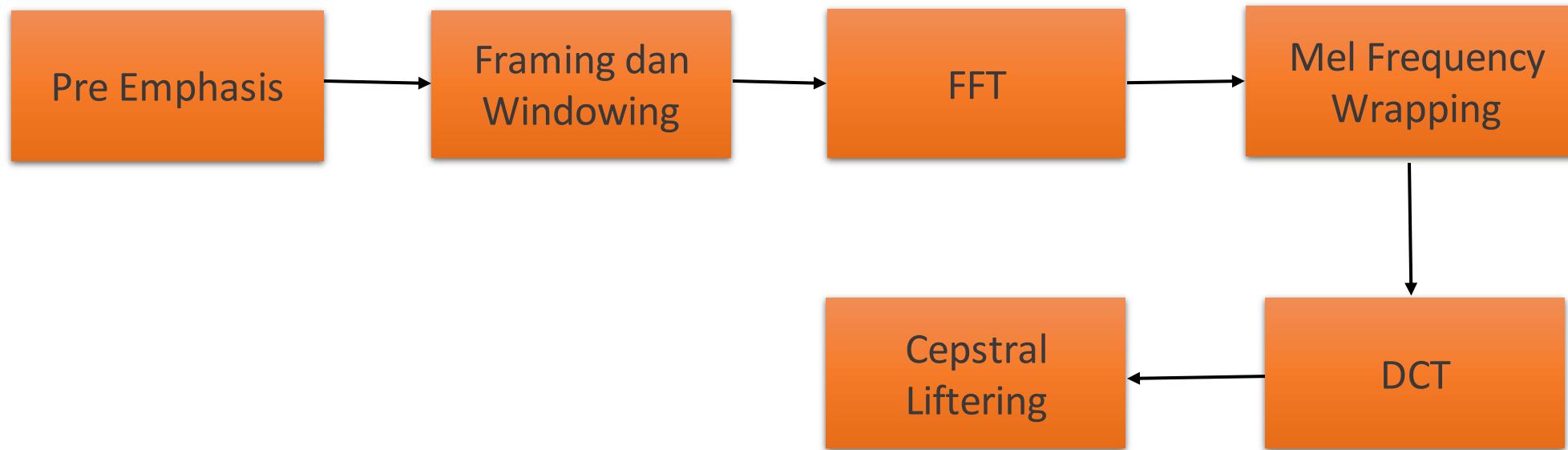
Mengubah ke sinyal diskrit terhadap waktu sehingga mendapatkan nilai biner

Contoh pada gambar 8 bit sehingga memiliki 256 level pada sinyal

Terakhir, tahap Encoding
Memasukan nilai biner setiap frame berurut sesuai waktu
Sehingga hasilnya urutan nilai biner 1 dan 0

Feature Extraction

Mel Frequency Cepstral Coefficient (MFCC)



Pre Emphasis

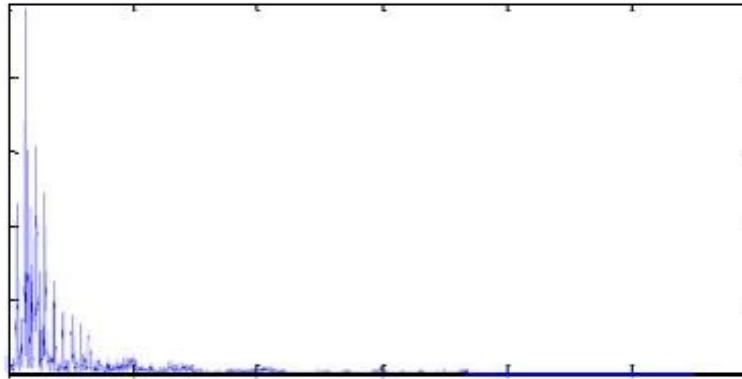
Memperkuat energi pada frekuensi. Sehingga membuat informasi lebih jelas

Dalam domain waktu

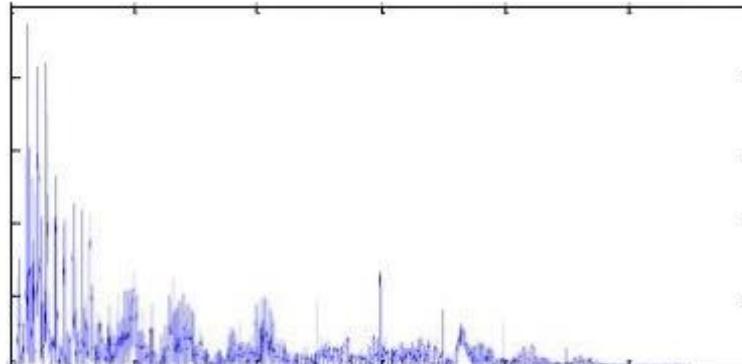
Input $x[n]$ dan $0.9 \leq \alpha \leq 1.0$,

$$y[n] = x[n] - \alpha x[n-1]$$

Sebelum



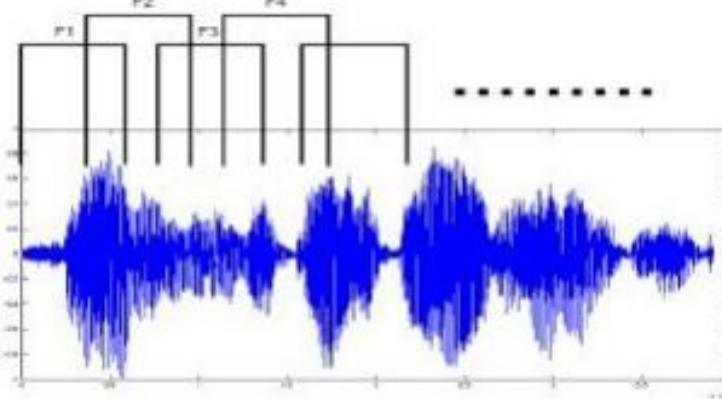
Sesudah



Framing dan Windowing

Framing proses untuk membagi sampel suara menjadi beberapa frame

Windowing mengurangi discontinuitas sinyal pada awal dan akhir bingkai setelah proses framing



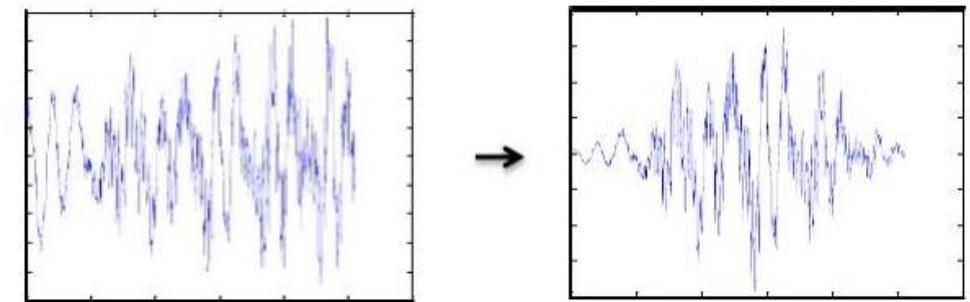
Ketika proses framing di samping terjadi tumpang tindih, sehingga dilakukan windowing

Dengan menggunakan Hamming Window

$$S_w(n) = \{0.54 - 0.46 \cos\left(\frac{2\pi(n-1)}{N-1}\right)\}, 1 \leq n \leq N$$

Dengan n waktu didapatkan hasil sinyal

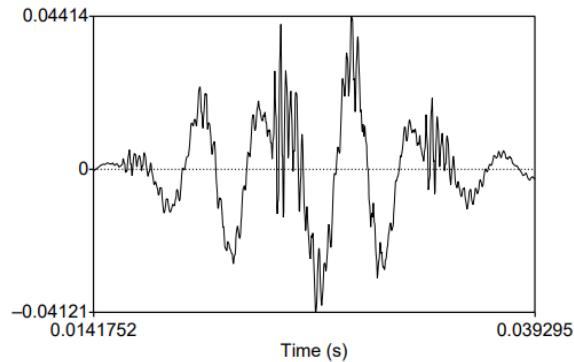
$$Y[n] = S_w[n] \times S_{frame}[n]$$



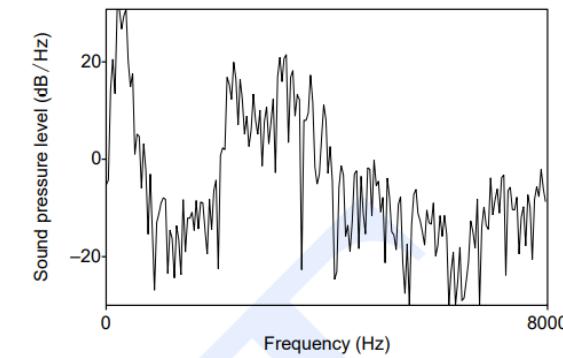
Fast Fourier Transform (FFT)

Mengubah domain
waktu menjadi
domain frekuensi

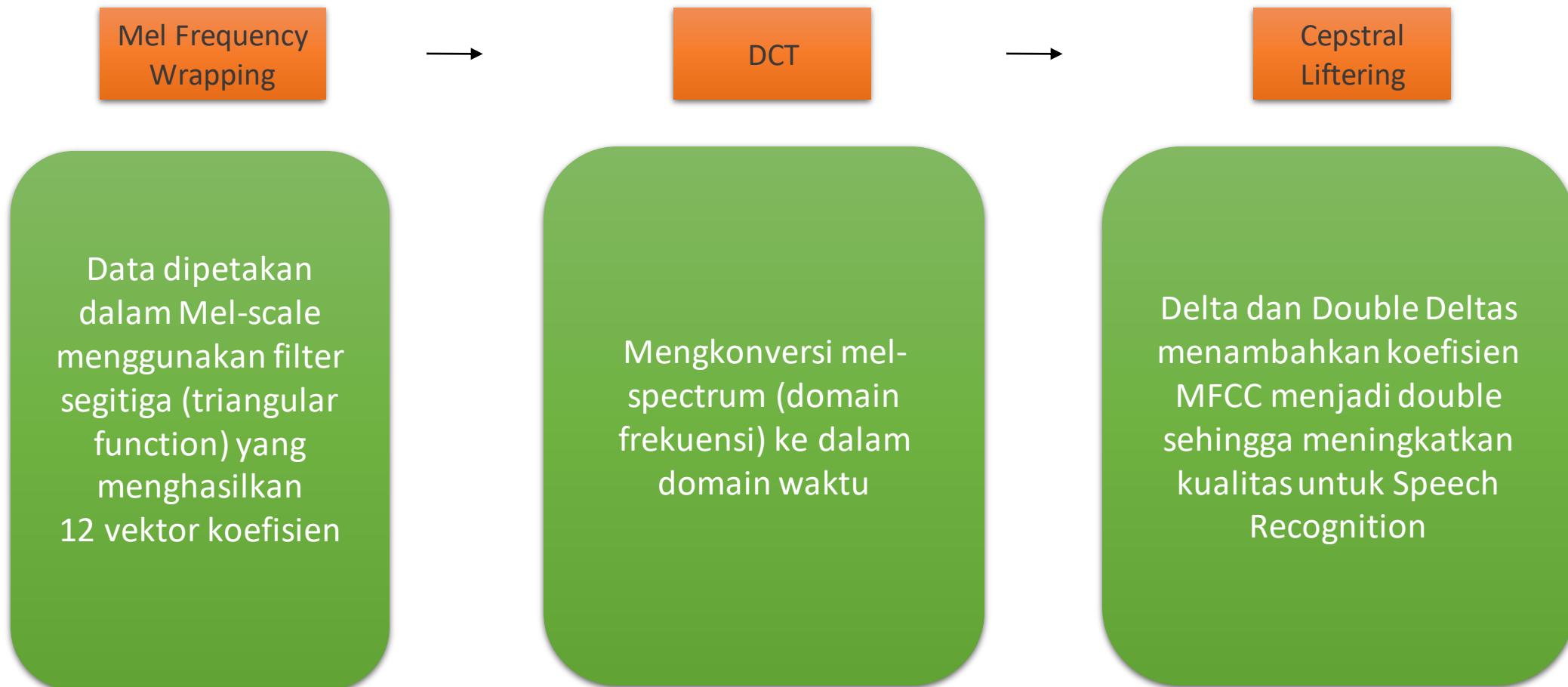
$$bin_k = \left| \sum_{n=1}^N S_w(n) e^{-i(n-1)k \frac{2\pi}{N}} \right|, k = 0, 1, 2, \dots, N-1$$

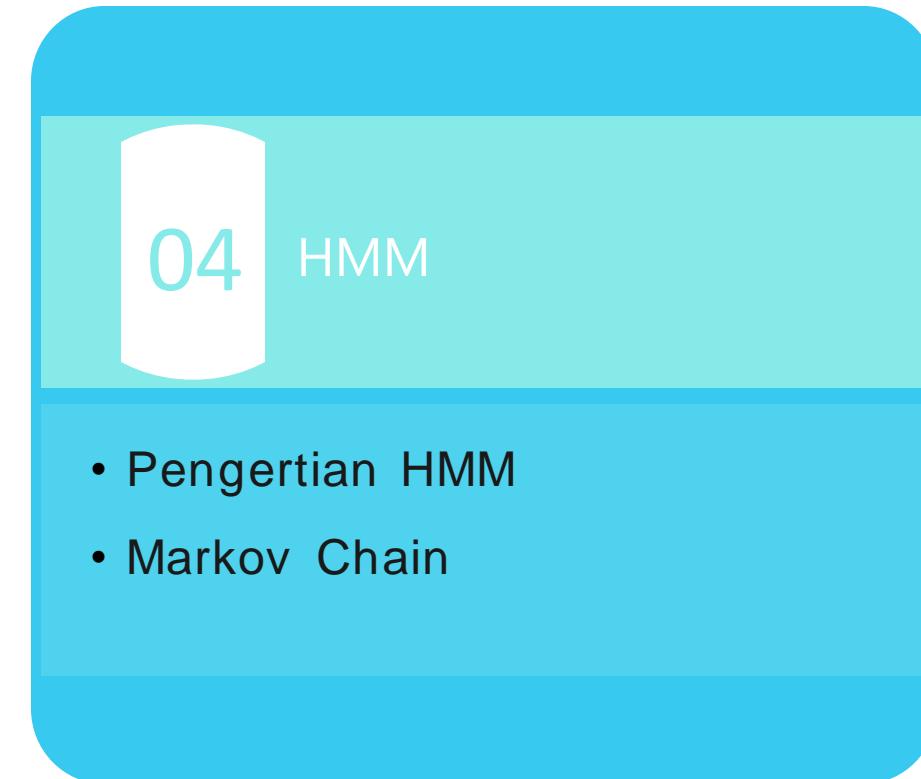
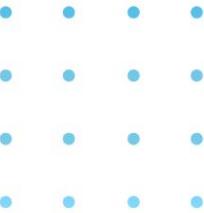


Sebelum (Time Domain)



Sesudah (Frequency Domain)





04 HMM

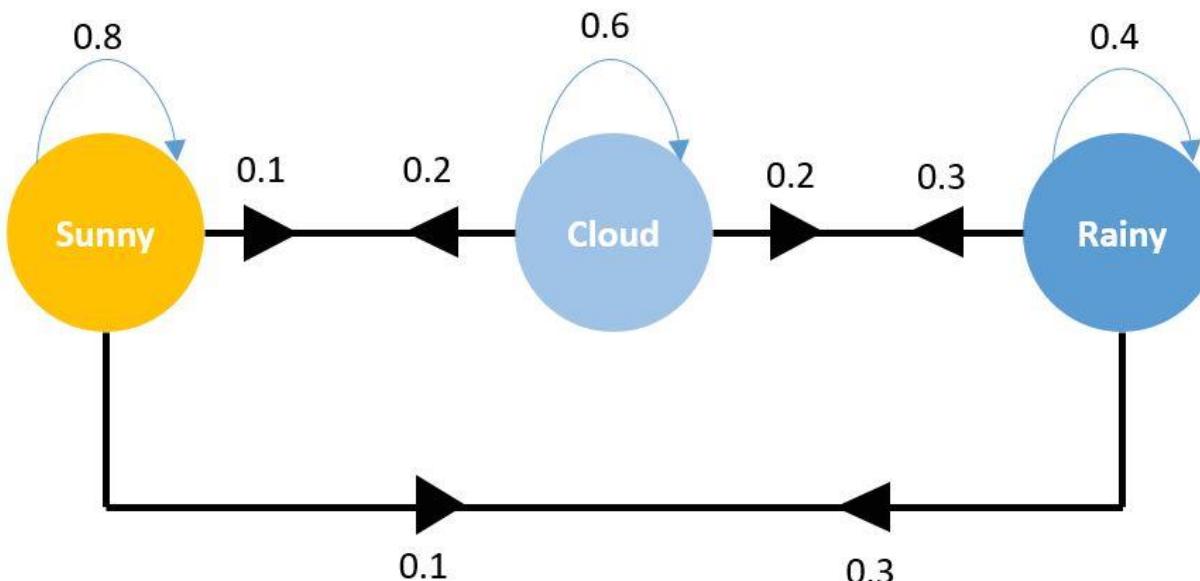
- Pengertian HMM
- Markov Chain



Markov Model

- Markov Chain (Markov Model) ditemukan oleh Andrey Markov. Ketika diberi inputan pada keadaan saat ini, maka keadaan yang akan datang dapat diprediksi

Markov Chain Transition Probabilities



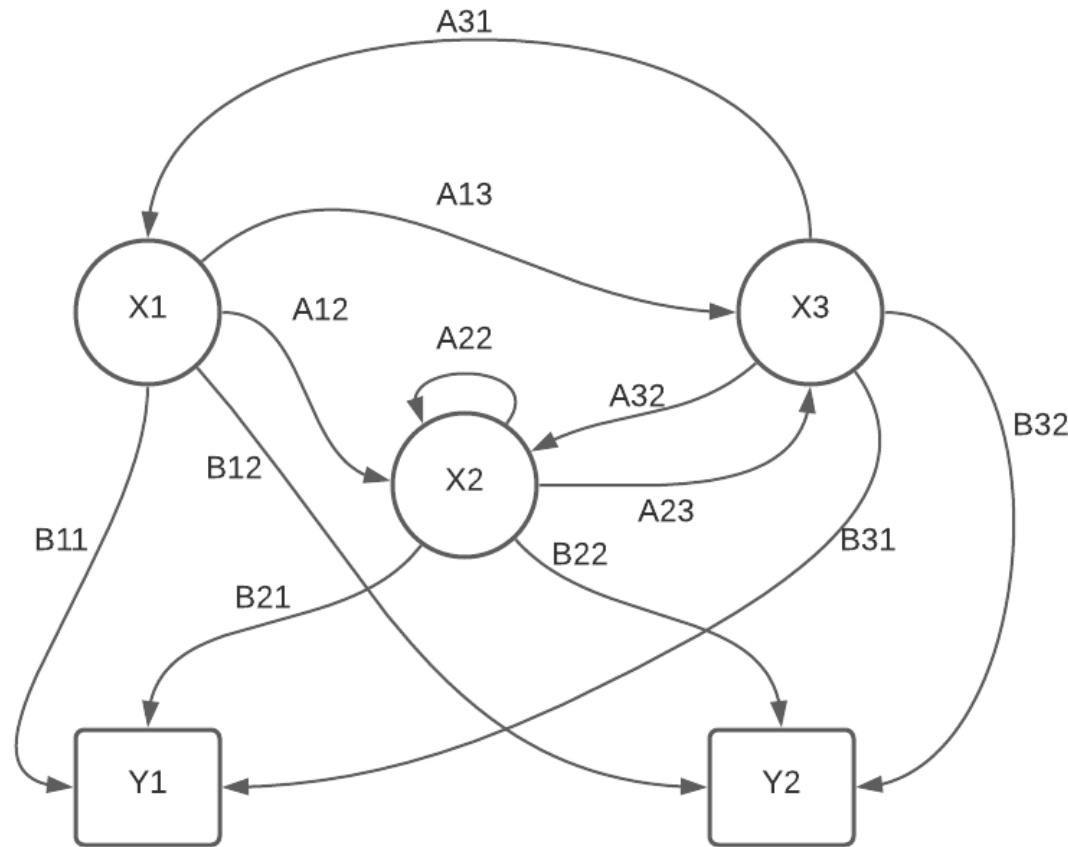
Matriks probabilitas dari kondisi disamping

| | Sunny | Cloudy | Raining |
|---------|-------|--------|---------|
| Sunny | 0.8 | 0.1 | 0.1 |
| Cloudy | 0.2 | 0.6 | 0.2 |
| Raining | 0.3 | 0.3 | 0.4 |

Hidden Markov Model

Markov Model tetapi memiliki kondisi yang tidak dapat diamati atau observasi, oleh sebab itu disebut hidden. Walaupun keadaan atau kondisi yang tidak dapat diamati, HMM memiliki output yang dapat terlihat

HMM



Dimana :

X = Kondisi (hidden)
Y = Output yang mungkin (Observable)
A = Probabilitas transisi kondisi
B = Probabilitas output

Misal Y adalah Bersyukur dan Tidak bersyukur, X adalah rejeki, musibah, normal

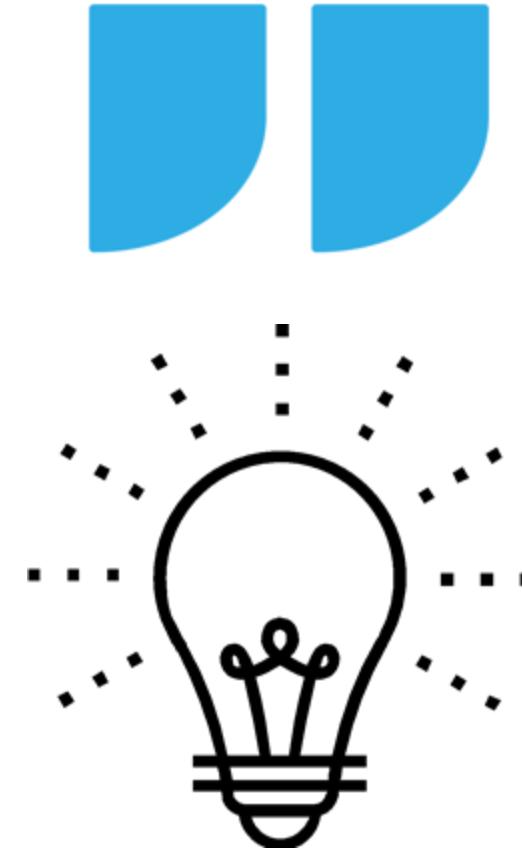


05 KESIMPULAN

- Ringkasan
- Kuis

Ringkasan

1. Pengertian Speech Recognition dan Perbedaan dengan Voice Recognition
2. Framework dan Tool dalam Speech Recognition
3. Proses Speech Recognition
4. Model dalam Speech Recognition
5. Tahapan mengubah sinyal analog ke digital
6. Proses Ekstrasi Fitur Audio
7. Pengertian Hidden Markov Model



Kuis

Pertanyaan

Yang tidak termasuk tahap mengubah sinyal analog ke digital?

- A. Sampling
- B. Windowing
- C. Kuantisasi
- D. Encoding



Kuis

Pertanyaan

Yang tidak termasuk tahap mengubah sinyal analog ke digital?

- A. Sampling
- B. Windowing**
- C. Kuantisasi
- D. Encoding



Jawaban B



TERIMA KASIH

Orbit Future Academy

PT Orbit Ventura Indonesia
Center of Excellence (Jakarta Selatan)
Gedung Veteran RI, Lt.15
Unit Z15-002, Plaza Semanggi
Jl. Jenderal Sudirman Kav.50, Jakarta
12930, Indonesia

- Jakarta Selatan/Pusat
- Jakarta Barat/BSD
- Kota Bandung
- Kab. Bandung
- Jawa Barat

Hubungi Kami

Director of Sales & Partnership
ira@orbitventura.com
+62 858-9187-7388

Social Media

-  [Orbit Future Academy](#)
-  [@OrbitFutureAcademyIn1](#)
-  [OrbitFutureAcademy](#)
-  [Orbit Future Academy](#)