Going Beyond the Basics: Advanced Deep Learning Techniques with TensorFlow

Gusti Triandi Winata

Community Researcher of RL & LLM
Cohere for AI

bangkit

About Me

Gusti Triandi Winata

Latest Work Experiences:

- Researcher, Cohere for AI **2024 - present**
- MLE Consultant, Freeport Indonesia **2023 - 2024**
- Mid. MLE, eFishery **2022 - 2023**
- OSS Fellowship with Adobe, Major League Hacking **2021**

Education:

- Bandung Institute of Technology Graduated at **2021**
  Bachelor of Electrical and Computer Engineering
  Was a Bangkit Graduate of the first batch (2020)!

bangkit

Ground Rules

Observe the following rules to ensure a supportive, inclusive, and engaging classes

Give full attention in class

Mute your microphone when you're not talking

Keep your camera on

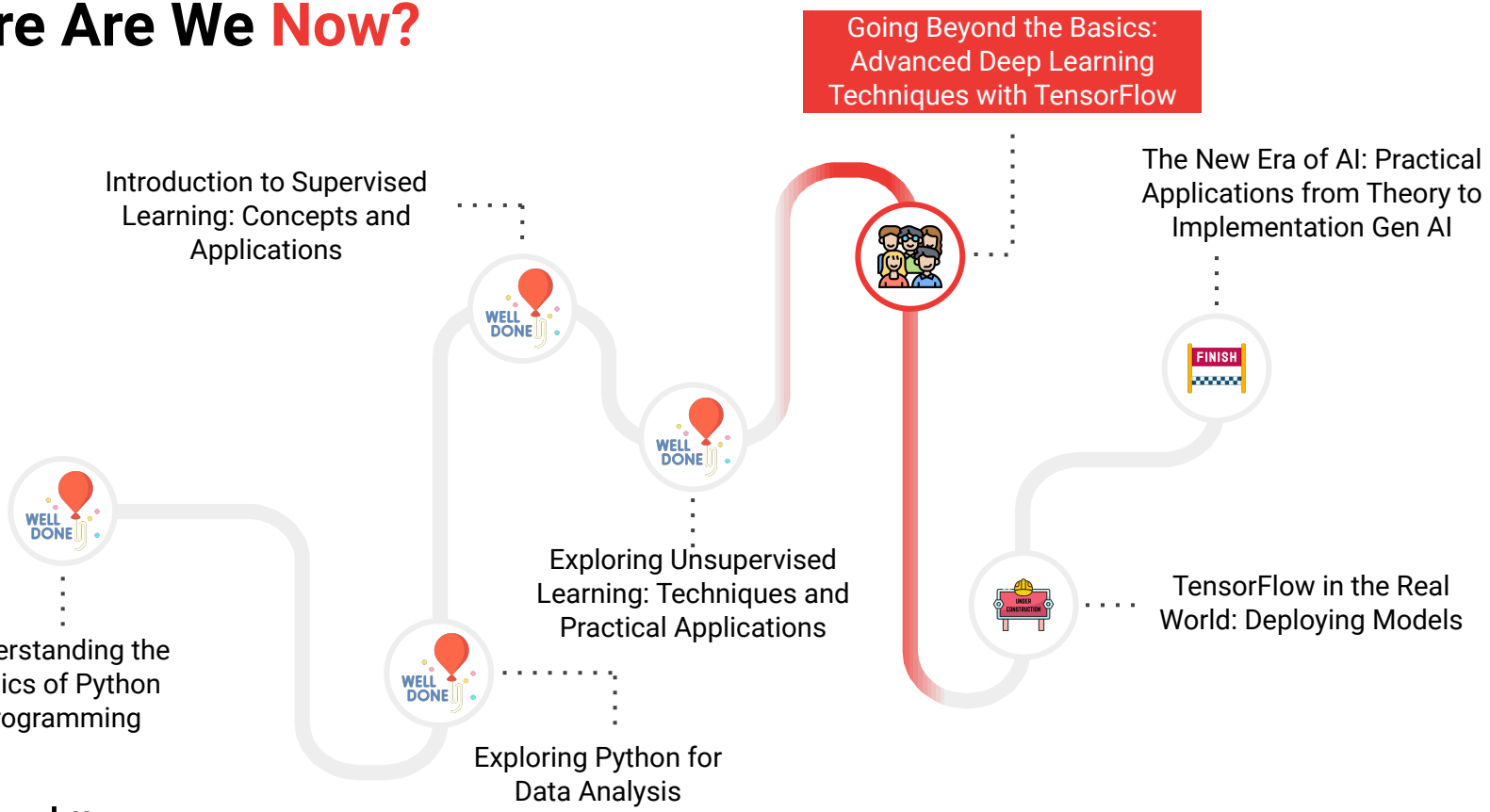
Turn on the CC Feature on Meet

Use raise hand or chat to ask questions

Make this room a safe place to learn and share

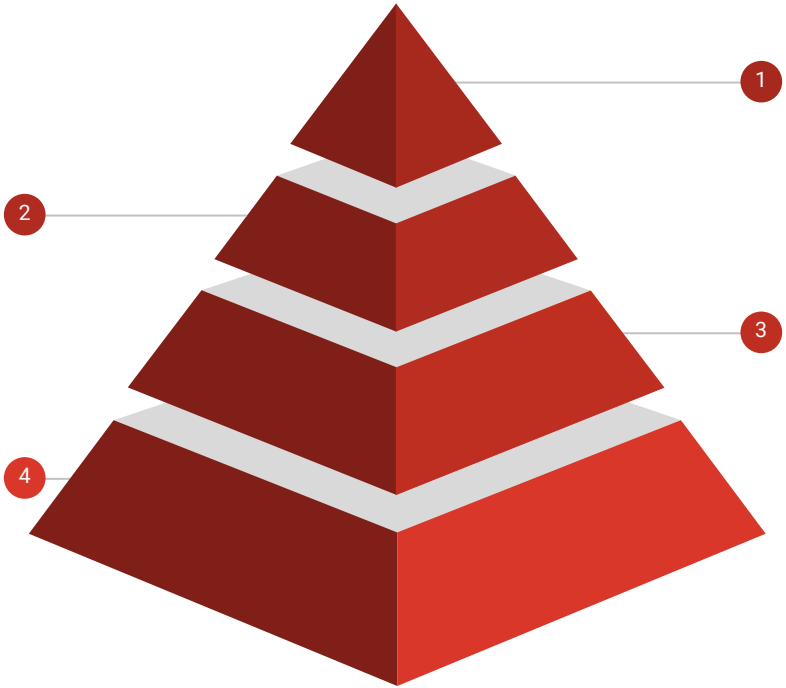
bangkit

Where Are We Now?



bangkit

Learning Objectives



1. Demonstrate how CNN Works
2. Examining Basic Concepts in NLP
3. Summarize How RNNs Work
4. Deciphering Basic Concepts for Time Series

Today's Agenda

1. Image Classification with Convolutional Neural Network

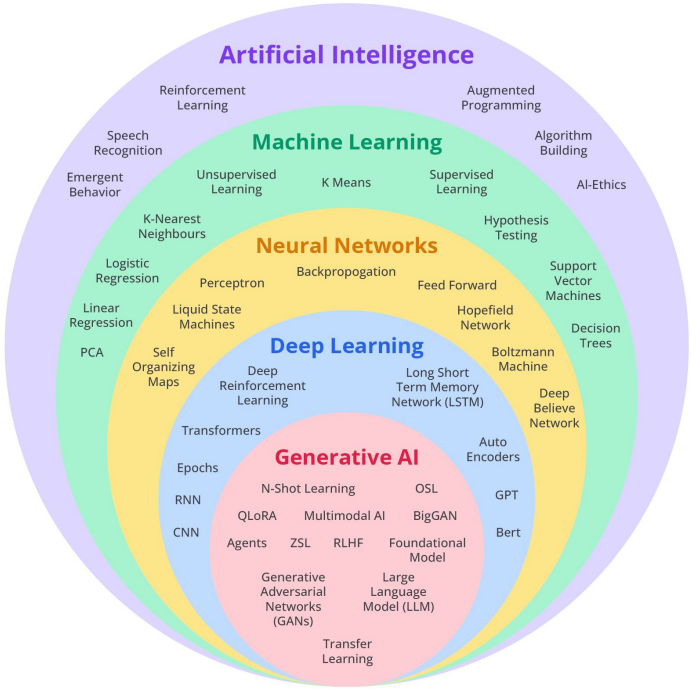
2. An Overview of Natural Language Processing

3. Exploring RNNs and LSTM

4. Introduction to Time Series Prediction

bangkit

Intermezzo



AI: Artificial Intelligence (AI) refers to the broad field of developing systems or machines that can mimic human intelligence to perform tasks such as decision-making, problem-solving, and understanding language.

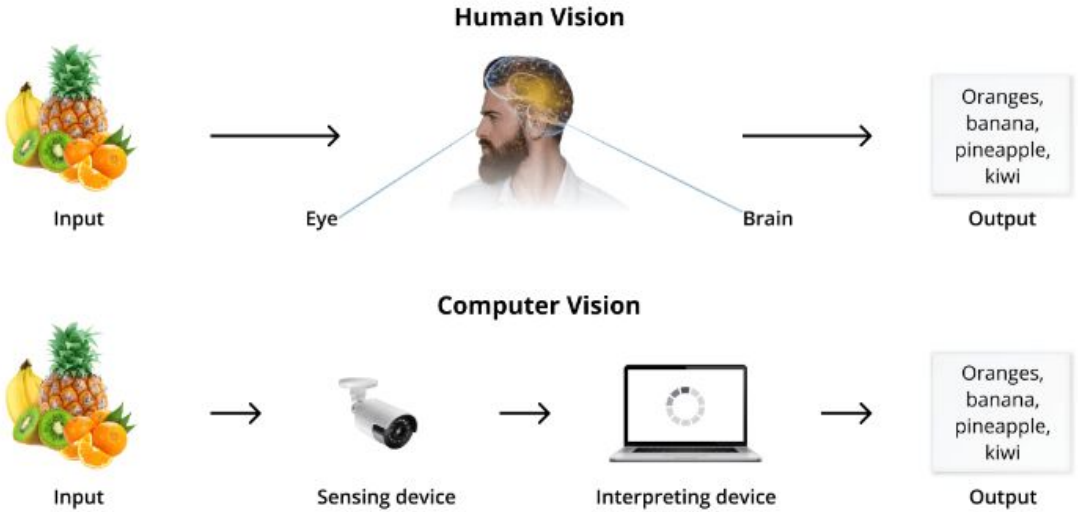
Machine Learning: Machine Learning (ML) is a subset of AI where computers learn patterns from data to make predictions or decisions without being explicitly programmed.

Deep Learning: Deep Learning is a specialized area of ML that uses neural networks with many layers to analyze complex patterns in large datasets, such as images or speech.

Generative AI: Generative AI focuses on creating new content like text, images, or music by learning patterns from existing data and generating outputs that mimic or extend those patterns.

bangkit

Human Vision VS Computer Vision



How Machine can Understand?

What humans see

```
08 02 22 97 38 15 00 40 00 75 01 05 07 78 52 12 50 77 91 03
19 49 99 40 17 81 18 57 60 67 17 40 98 43 69 45 04 56 62 00
61 49 31 73 55 79 14 29 93 71 10 67 53 88 30 03 19 13 36 65
52 70 96 23 04 60 11 42 69 24 65 66 01 32 56 71 37 02 36 92
22 31 16 71 51 67 43 00 41 92 36 54 22 40 40 25 66 33 13 60
24 47 32 60 99 03 45 02 44 75 33 55 70 36 04 20 35 17 12 50
32 90 61 28 64 23 47 10 26 35 40 47 59 54 70 66 10 30 44 70
67 26 20 68 02 62 12 20 95 63 94 39 63 08 40 91 66 49 94 22
24 55 56 05 66 73 99 26 97 17 70 70 96 83 14 00 34 09 43 72
21 36 23 09 75 00 76 44 20 45 35 14 00 61 33 97 34 31 33 95
70 17 53 25 22 75 31 67 15 94 03 80 40 62 16 14 09 53 56 92
16 39 06 42 96 35 31 47 55 53 88 24 00 17 54 24 36 29 85 57
26 36 00 48 55 71 69 07 05 44 44 37 44 60 21 55 51 54 17 55
19 60 01 65 05 94 47 69 20 73 92 13 86 52 17 77 04 89 55 40
04 52 02 03 97 35 99 16 07 97 57 32 16 26 26 79 33 27 98 66
00 36 63 07 57 62 20 72 03 46 33 67 46 55 22 32 63 93 53 69
04 42 16 73 38 25 39 11 24 94 72 10 08 46 29 32 40 42 76 36
20 49 36 41 72 30 23 00 34 62 99 69 02 67 59 05 74 04 36 16
20 73 33 29 78 31 90 01 74 31 49 71 48 86 01 16 23 57 05 54
01 70 54 71 03 51 54 69 16 92 33 48 61 43 52 01 89 19 67 48
```

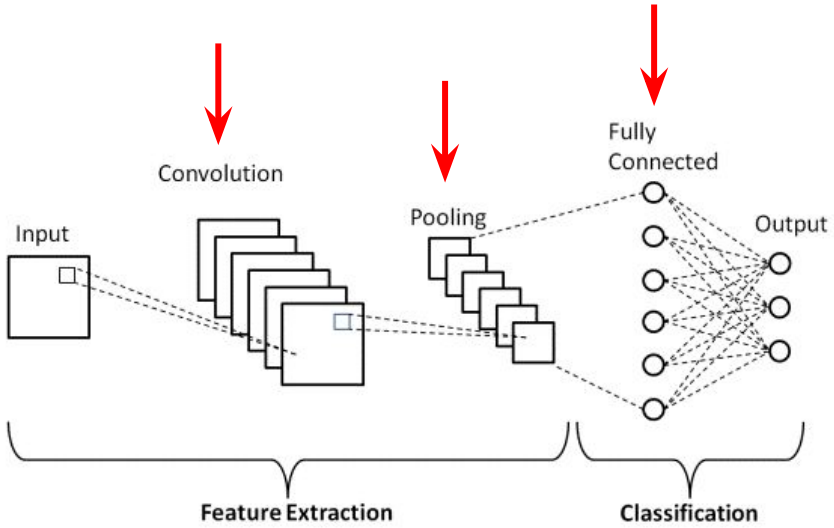
What the computer sees

Image Classification with Convolutional Neural Network

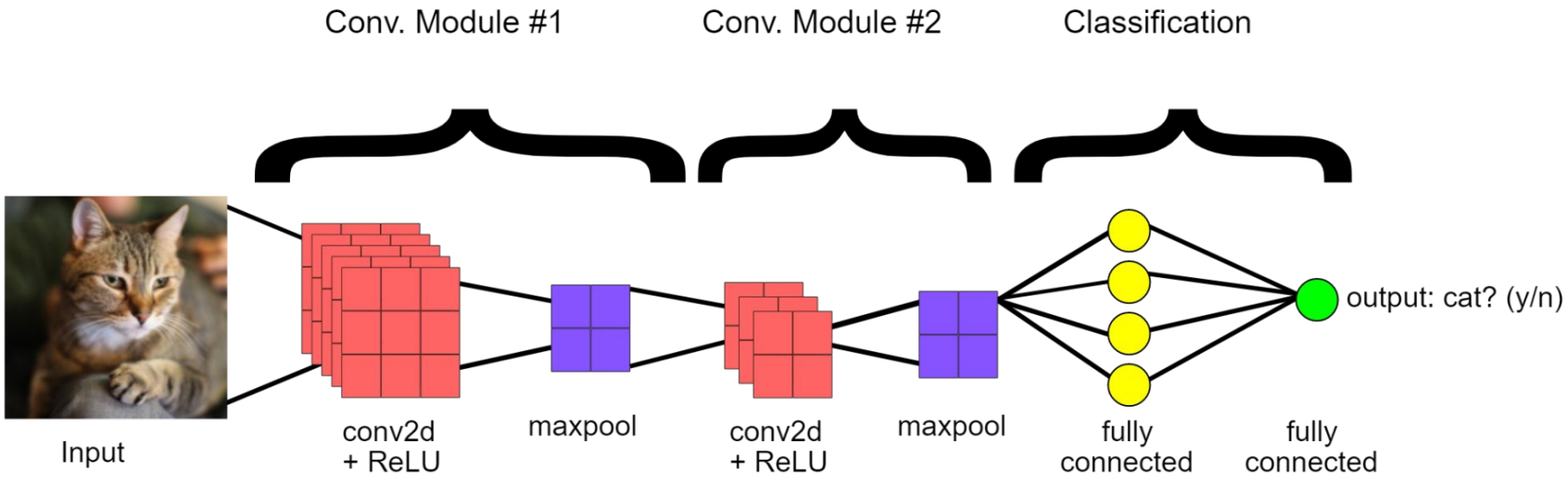
bangkit

What is Convolutional Neural Network?

CNN is a type of neural network model which allows us to extract higher representations for the image content.



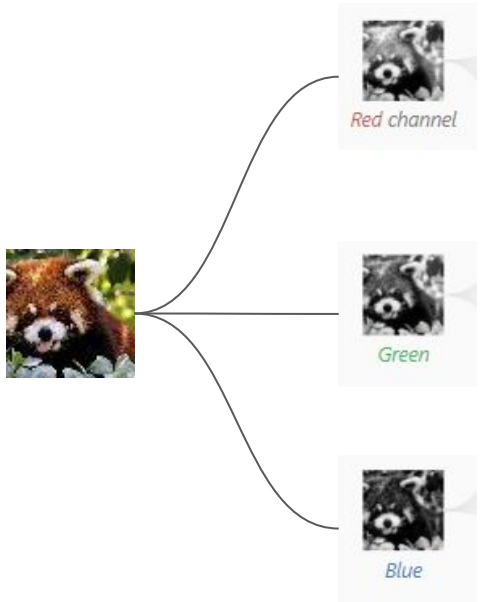
Typical CNN Architecture



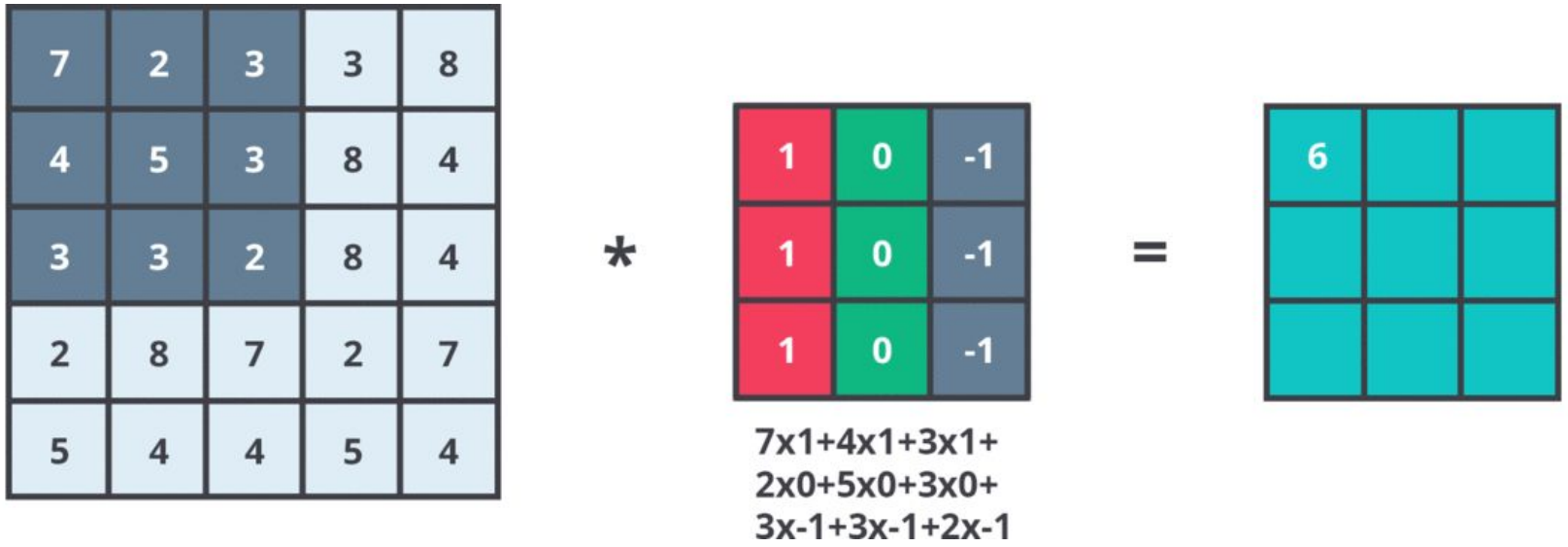
Shape: (32, 32, 3)

Typical CNN Architecture - Input Layer

The input layer functions to receive input data or input from the dataset that will be processed by the network.



Typical CNN Architecture - Convolution Layer

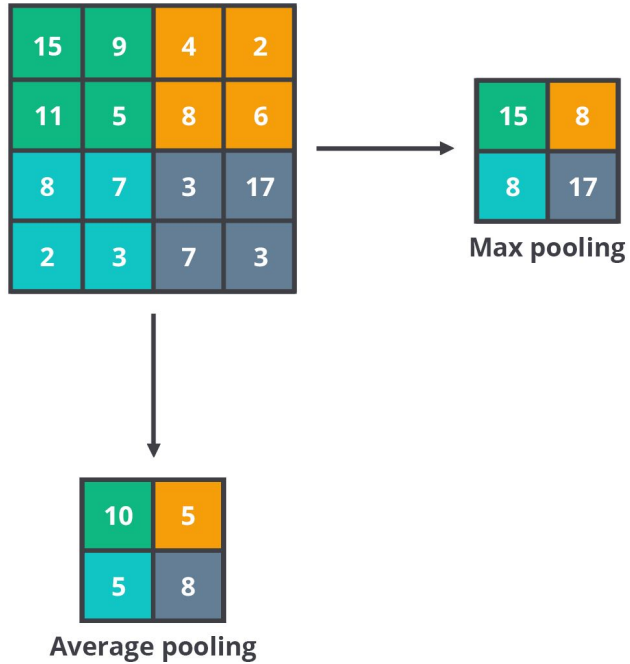


Conv2D: https://www.tensorflow.org/api_docs/python/tf/keras/layers/Conv2D

Typical CNN Architecture - Pooling Layer

Pooling is a type of downsampling that often occurs after convolution.

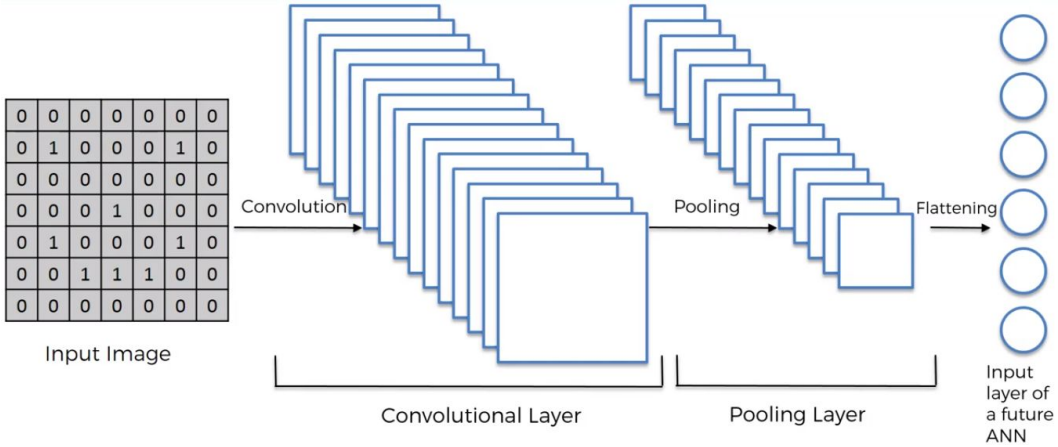
The goal is to reduce the size of the training data before it goes into the fully connected network without losing much information.



MaxPooling2D: https://www.tensorflow.org/api_docs/python/tf/compat/v1/layers/MaxPooling2D

Typical CNN Architecture - Flatten Layer

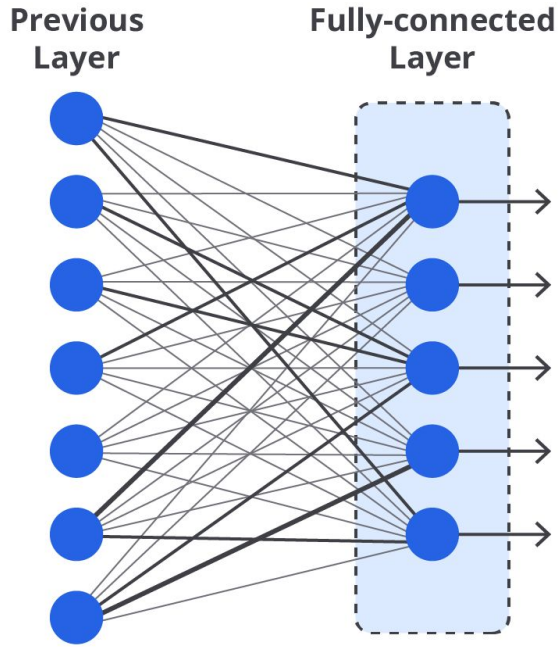
The Flatten layer **reduces the dimensionality** of the data, which can help reduce the number of parameters in the subsequent fully connected layers.



Typical CNN Architecture - Fully Connected Layers

Fully connected layers are an important part of a neural network located at the end of the architecture, usually after a series of convolution layers and pooling layers in a CNN.

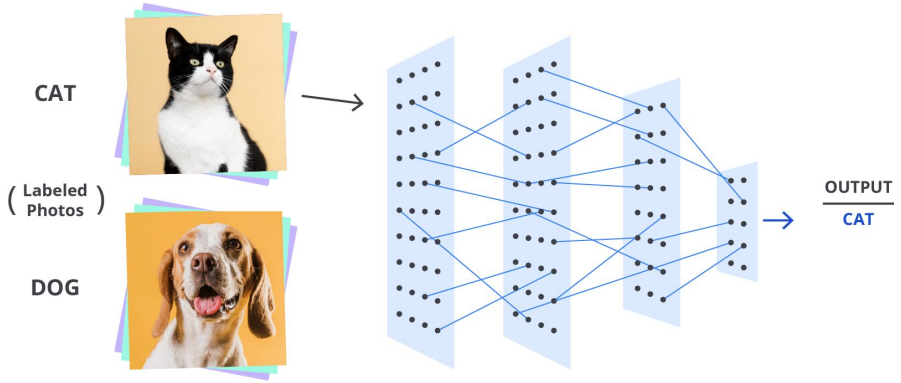
The function of this layer is to combine all the features extracted from the previous layers to make a final decision, such as classification or prediction.



Typical CNN Architecture - Output Layers

Output layers, the last part, produce the predictions or final output of the model.

The function of this layer is highly dependent on the type of task being performed by the network.



How to build CNN Architecture?

```
### START CODE HERE ###
# Retrieve dimensions from A_prev's shape (≈1 line)
(m, n_H_prev, n_W_prev, n_C_prev) = A_prev.shape

# Retrieve dimensions from W's shape (≈1 line)
(f, f, n_C_prev, n_C) = W.shape

# Retrieve information from "hparameters" (≈2 lines)
stride = hparameters['stride']
pad = hparameters['pad']

# Compute the dimensions of the CONV output volume using the formula given above. Hint: use int() to floor. (≈2 lines)
n_H = int((n_H_prev - f + 2 * pad) / stride) + 1
n_W = int((n_W_prev - f + 2 * pad) / stride) + 1

# Initialize the output volume Z with zeros. (≈1 line)
Z = np.zeros((m, n_H, n_W, n_C))

# Create A_prev_pad by padding A_prev
A_prev_pad = zero_pad(A_prev, pad)

for i in range(m):                               # loop over the batch of training examples
    a_prev_pad = A_prev_pad[i]                               # Select ith training example's padded activation
    for h in range(n_H):                         # loop over vertical axis of the output volume
        for w in range(n_W):                     # loop over horizontal axis of the output volume
            for c in range(n_C):                 # loop over channels (= #filters) of the output volume

                # Find the corners of the current "slice" (≈4 lines)
                vert_start = h * stride
                vert_end = vert_start + f
                horiz_start = w * stride
                horiz_end = horiz_start + f

                # Use the corners to define the (3D) slice of a_prev_pad (See Hint above the cell). (≈1 line)
                a_slice_prev = a_prev_pad[vert_start:vert_end, horiz_start:horiz_end, :]

                # Convolve the (3D) slice with the correct filter W and bias b, to get back one output neuron. (≈1 line
                Z[i, h, w, c] = conv_single_step(a_slice_prev, W[...,c], b[...,c])
```

Convolutional

```
# Retrieve dimensions from the input shape
(m, n_H_prev, n_W_prev, n_C_prev) = A_prev.shape

# Retrieve hyperparameters from "hparameters"
f = hparameters["f"]
stride = hparameters["stride"]

# Define the dimensions of the output
n_H = int(1 + (n_H_prev - f) / stride)
n_W = int(1 + (n_W_prev - f) / stride)
n_C = n_C_prev

# Initialize output matrix A
A = np.zeros((m, n_H, n_W, n_C))

### START CODE HERE ###
for i in range(m):                               # loop over the training examples
    for h in range(n_H):                           # loop on the vertical axis of the output volume
        for w in range(n_W):                       # loop on the horizontal axis of the output volume
            for c in range (n_C):                  # loop over the channels of the output volume

                # Find the corners of the current "slice" (≈4 lines)
                vert_start = h * stride
                vert_end = vert_start + f
                horiz_start = w * stride
                horiz_end = horiz_start + f

                # Use the corners to define the current slice on the ith training example of A_prev, channel c. (≈1
                a_prev_slice = A_prev[i, vert_start:vert_end, horiz_start:horiz_end, c]

                # Compute the pooling operation on the slice. Use an if statment to differentiate the modes. Use np
                if mode == "max":
                    A[i, h, w, c] = np.max(a_prev_slice)
                elif mode == "average":
                    A[i, h, w, c] = np.mean(a_prev_slice)

### END CODE HERE ###
```

Max Pooling

How to build CNN Architecture?

[Hands-on] Unlocking the Power of CNNs for Image Classification

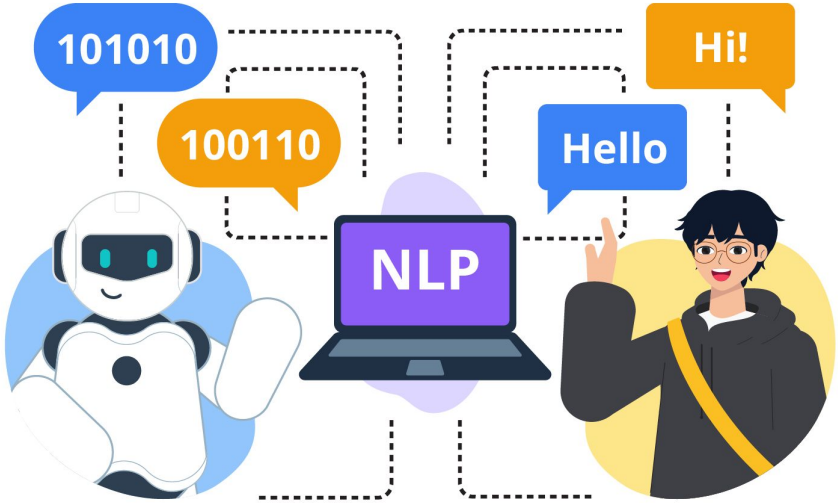
bangkit

An Overview of Natural Language Processing

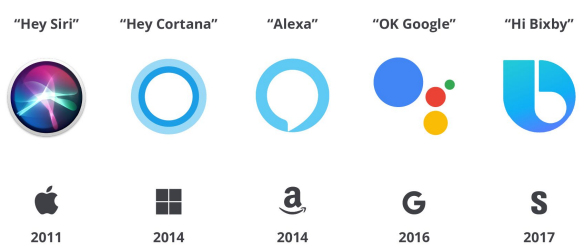
bangkit

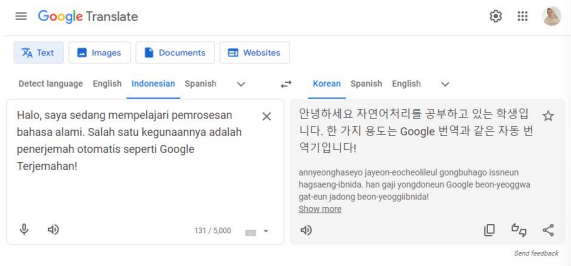
What is NLP?

Natural Language Processing (NLP) is a branch of Artificial Intelligence that gives machines the ability to **read**, **process** and **derive meaning** from human languages



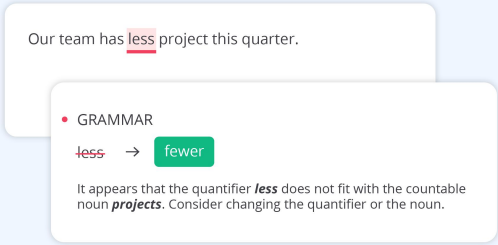
The Applications of NLP in Daily Life

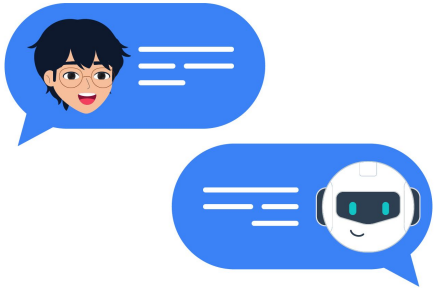




Sentiment Analysis

Menemukan opini, emosi, dan perasaan pengguna tentang suatu produk atau layanan





bangkit

How Computer Processing A Text

I love my cat

{ “I” : 1, “love” : 2, “my” : 3, “cat” : 4 }

Encoded Word = [1, 2, 3, 4]

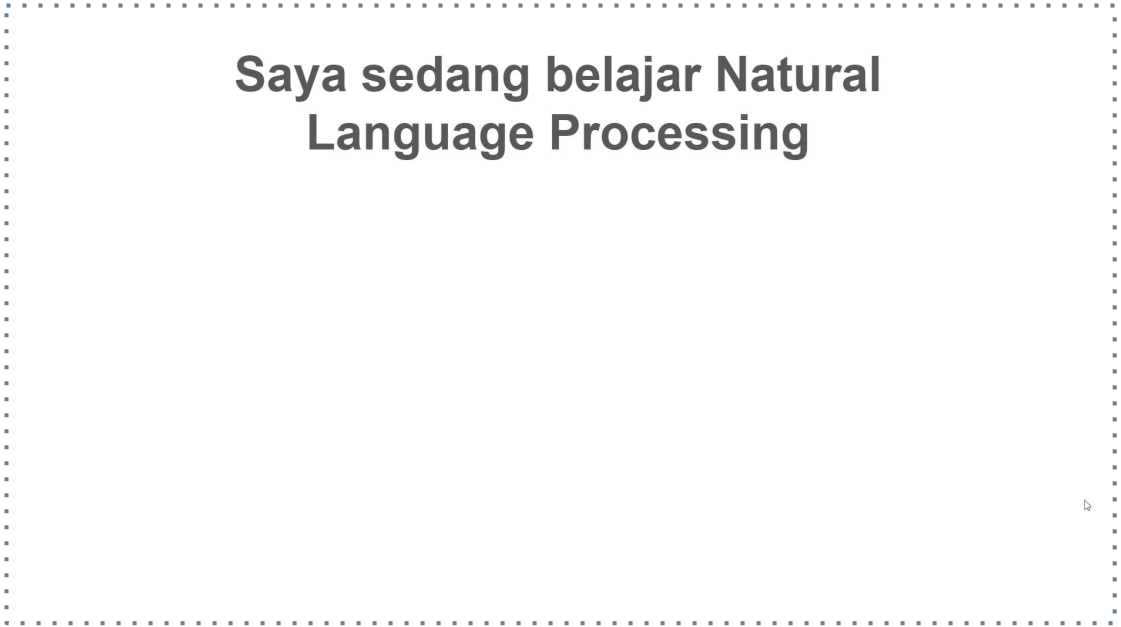
```python
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.preprocessing.text import Tokenizer

sentences = ["I love my cat"]

tokenizer = Tokenizer(num_words=100)
tokenizer.fit_on_texts(sentences)
sequences = tokenizer.texts_to_sequences(sentences)

print(tokenizer.word_index)
print(sequences)
```

bangkit

Tokenizer: https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/text/Tokenizer

Additional Tips for How Computer Processing A Text

Case folding aims to **convert** all letters in a text document **into lowercase letters**.



Additional Tips for How Computer Processing A Text

Removal special characters aims to **rid the text of unwanted special characters**, such as numbers, punctuation marks, symbols, or other special characters.

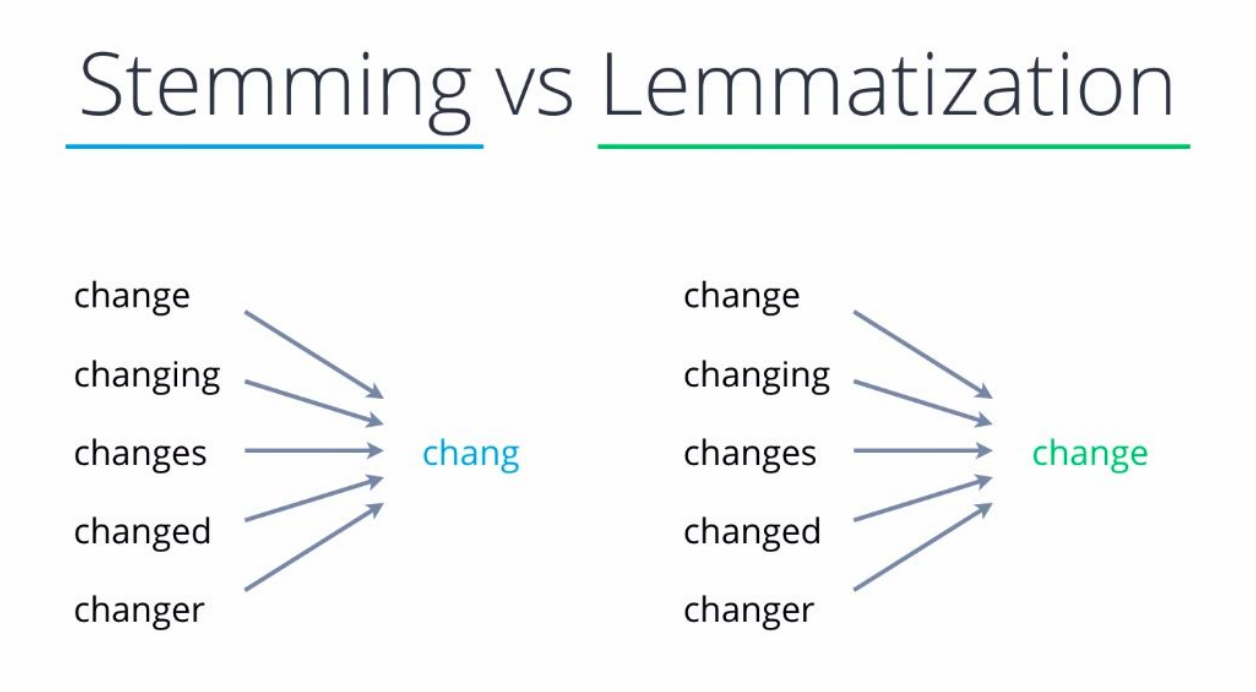
@dicoding: Hallo semuanya!! Masih semangat belajar NLP kan??! Berapa persen semangatnya? 50%? 60%? atau 100%? #menyalaabangkuh! >.<

Additional Tips for How Computer Processing A Text

Stopwords removal (filtering) is the **removal of stopwords or common words** that appear frequently, but do not have high informative value in text analysis.

Makanan favorit adik saya adalah nasi goreng dan ayam goreng crispy karena sangat enak dan menggugah selera

Additional Tips for How Computer Processing A Text



Additional Tips for How Computer Processing A Text

```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.preprocessing.text import Tokenizer

sentences = [
    "I love my cat",
    "I love my dog",
    "Do you think my dog is amazing?",
    "Do you think my cat is cute?"
]

tokenizer = Tokenizer(num_words=100, oov_token="<OOV>")
tokenizer.fit_on_texts(sentences)
sequences = tokenizer.texts_to_sequences(sentences)

print(tokenizer.word_index)
print(sequences)
```

Put a special value in when an unseen word encountered.

[[3, 4, 2, 5], [3, 4, 2, 6], [7, 8, 9, 2, 6, 10, 11], [7, 8, 9, 2, 5, 10, 12]]

Padding

Padding is used to fit all dictionaries in one sequence. So, all sequences have the same length.

```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences

sentences = [
    "I love my cat",
    "I love my dog",
    "Do you think my dog is amazing?",
    "Do you think my cat is cute?"
]

tokenizer = Tokenizer(num_words=100, oov_token="<OOV>")
tokenizer.fit_on_texts(sentences)
sequences = tokenizer.texts_to_sequences(sentences)

padded = pad_sequences(sequences, padding='post', truncating='post', maxlen=10 )

print(tokenizer.word_index)
print(padded)
```

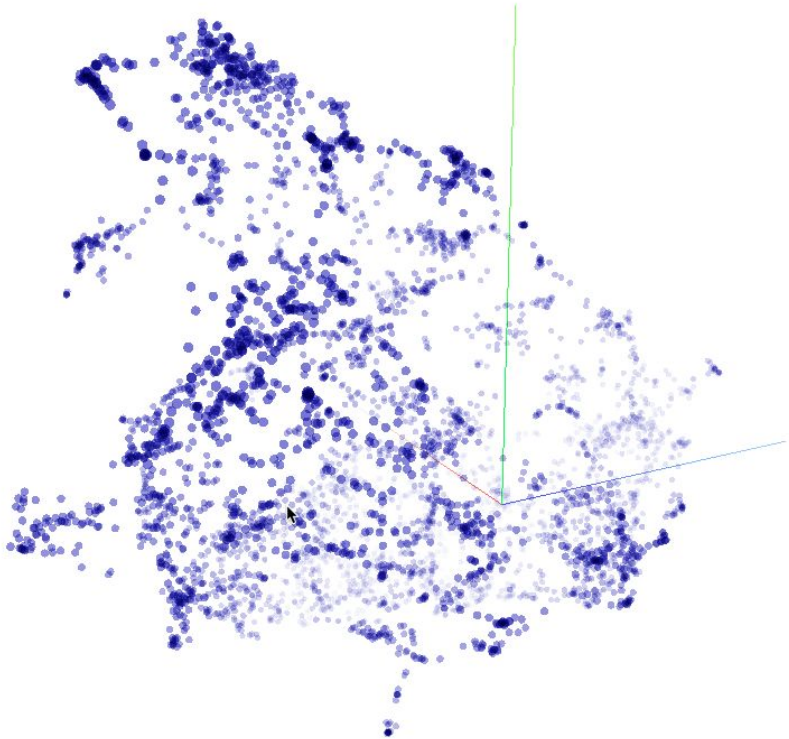
```
[[ 3  4  2  5  0  0  0  0  0  0]
 [ 3  4  2  6  0  0  0  0  0  0]
 [ 7  8  9  2  6 10 11  0  0  0]
 [ 7  8  9  2  5 10 12  0  0  0]]
```

bangkit

pad_sequences: https://www.tensorflow.org/api_docs/python/tf/keras/utils/pad_sequences

Word Embeddings

Word embeddings are a type of word representation that allows words with similar meanings to have similar representations.

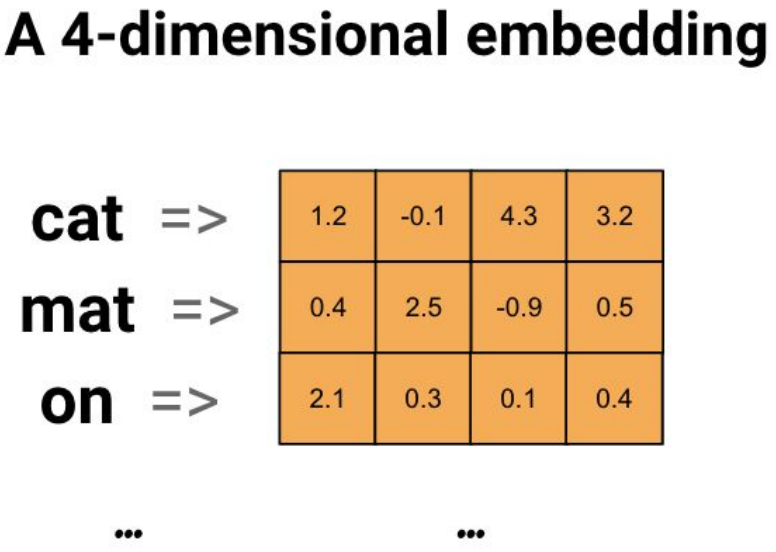


bangkit

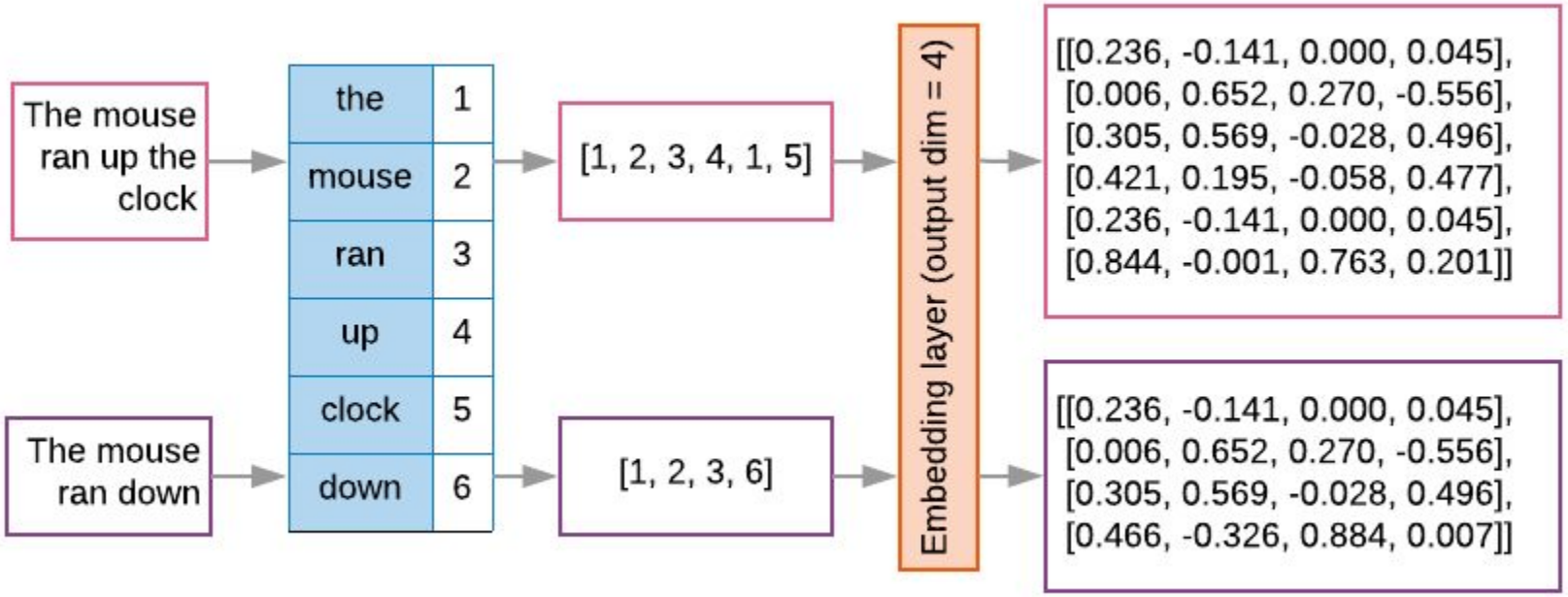
TensorFlow Embedding Projector: https://projector.tensorflow.org/

Word Embedding

An embedding is a dense vector of floating point values (the length of the vector is a parameter you specify).



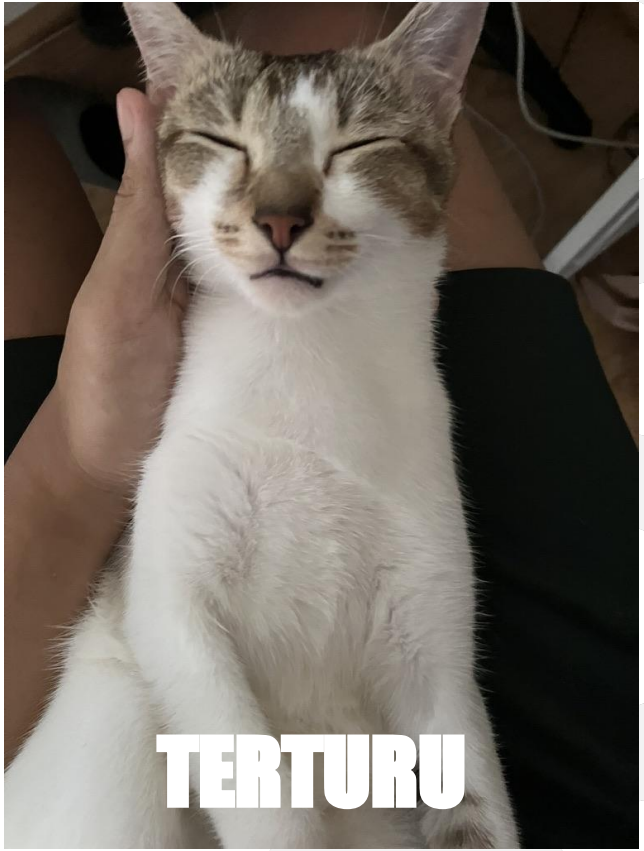
Word Embedding



[Hands-on] Riding the Wave of NLP Experience

bangkit

Break Time!



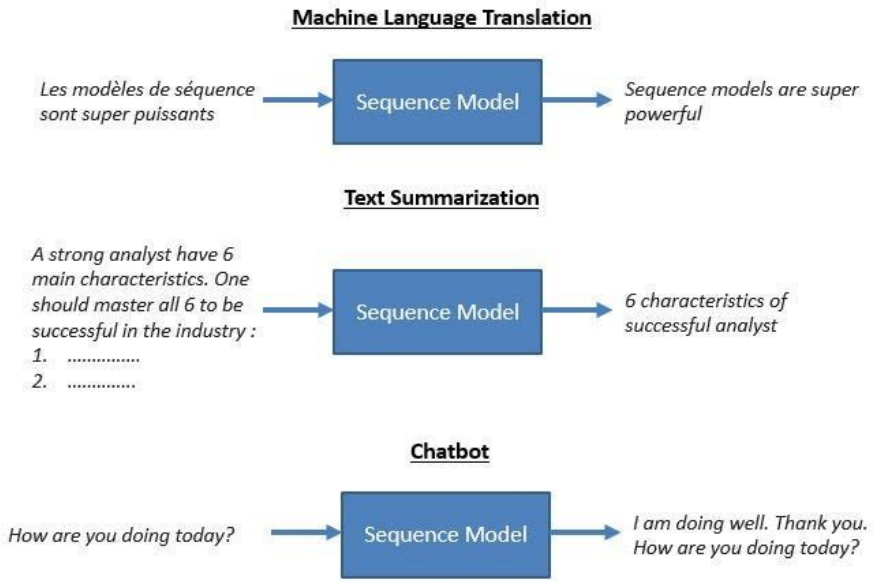
bangkit

Exploring RNNs and LSTM

bangkit

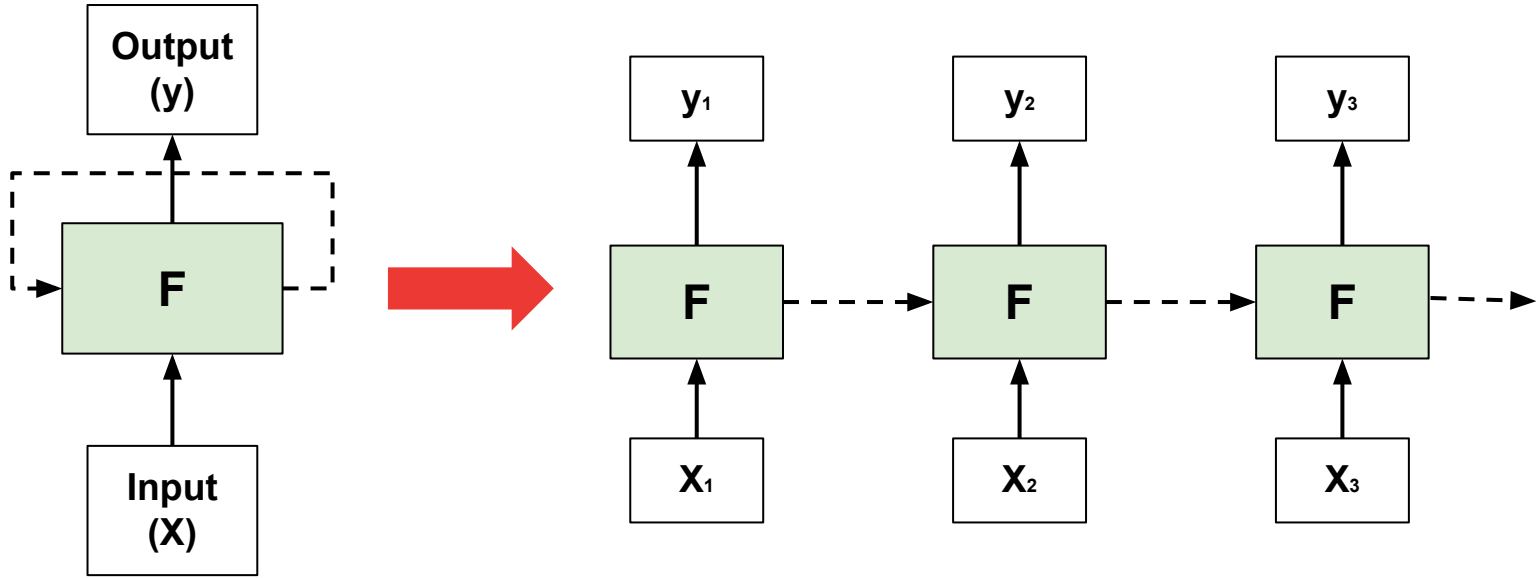
Limitation of Basic ANN Model

- In sequential problems, the **order of values matters**.
- To handle this, the model needs to **use previous experience** to process the **next input value**.
- Basic ANN model **can't do this**.



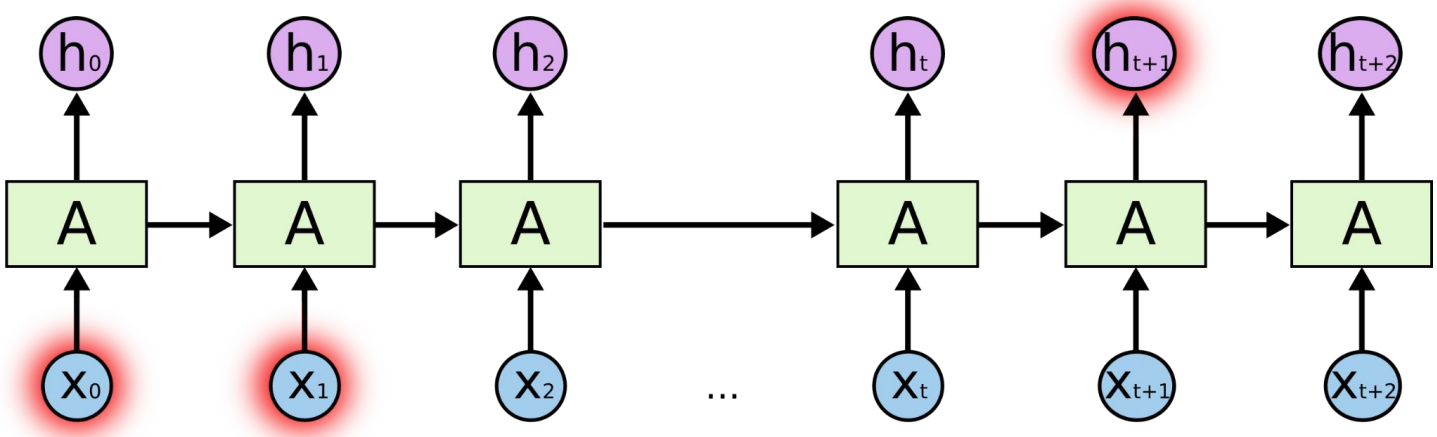
RNN Model Architecture

A recurrent neural network (RNN) algorithm is a type of deep learning algorithm designed to process sequence data.



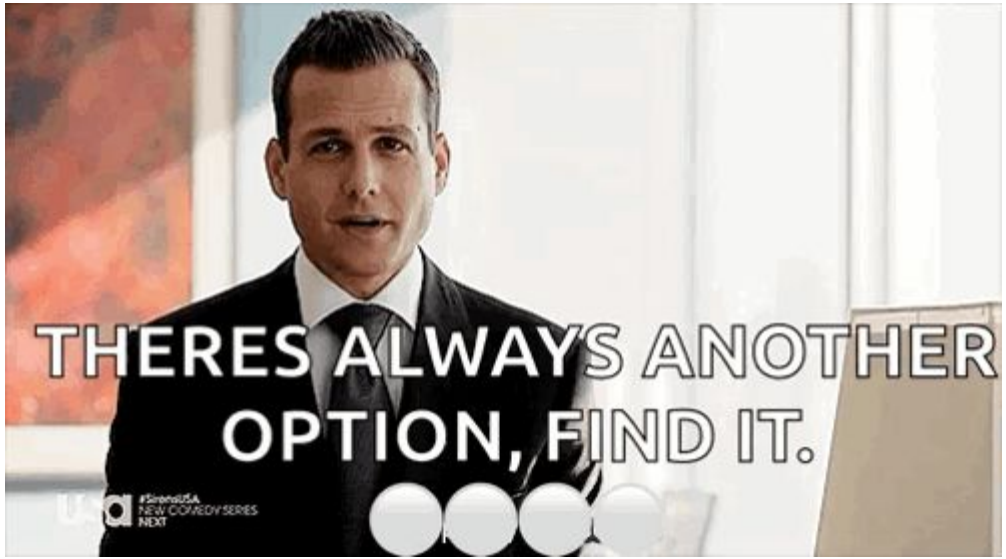
bangkit

The Downside of RNN Models



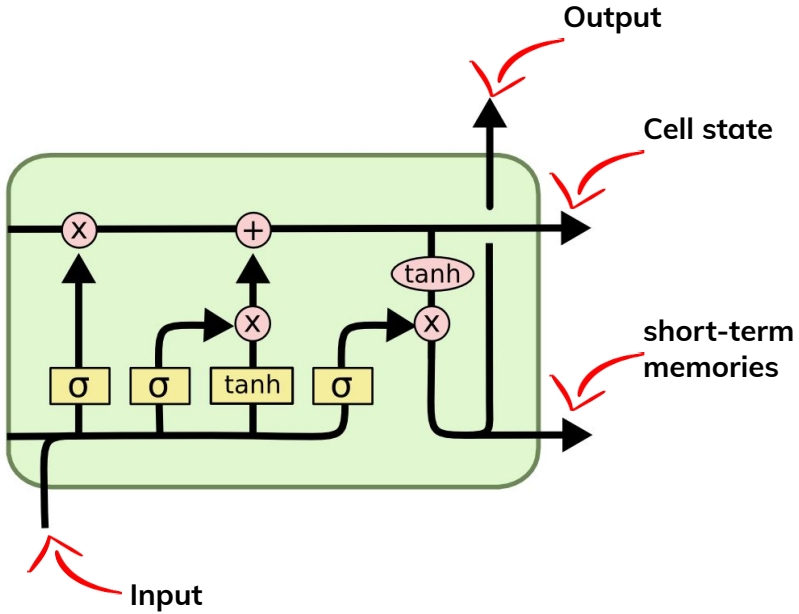
RNN never really did well at handling large sequences of data, like long paragraphs. By the time they were process the end of a paragraph they'd forget what happened in the beginning.

Is it the only option to solve the problem?

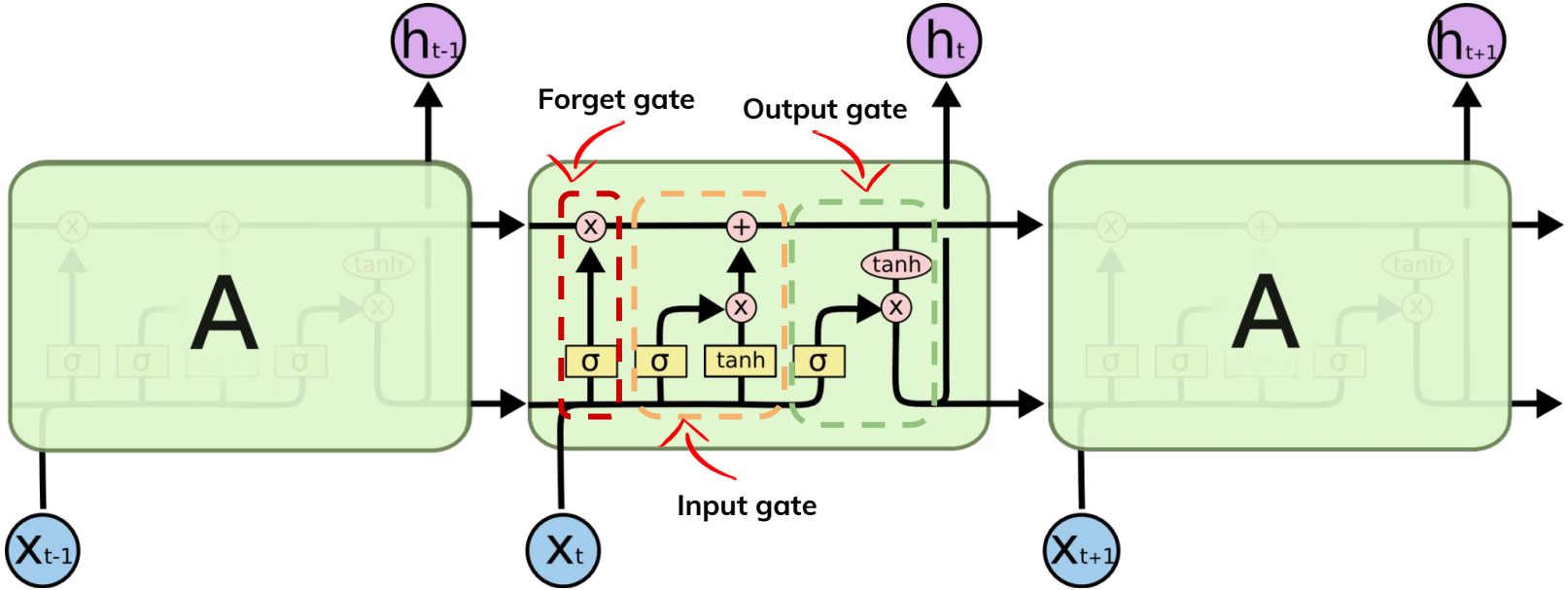


Long Short Term Memory Network

- LSTMs are a **special kind of RNN**, capable of learning long-term dependencies.
- LSTM have **long-term memories called cell state** to handle long-term dependencies.



LSTM Model Architecture

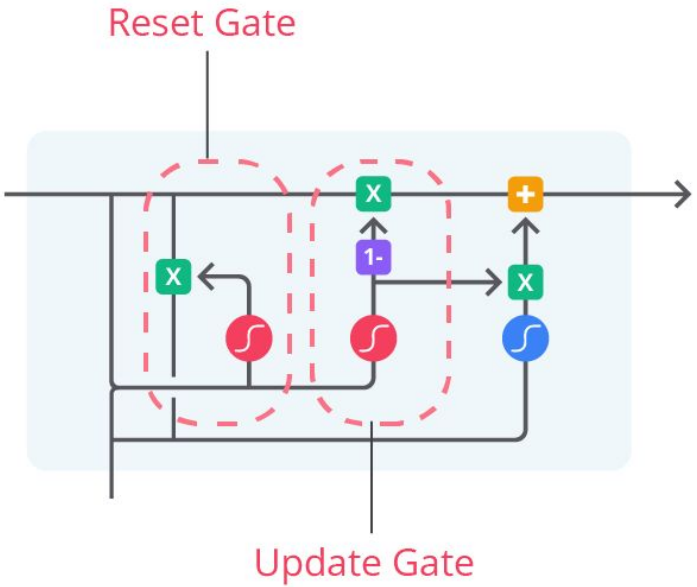


bangkit

Gated Recurrent Unit (GRU)

GRUs are a type of unit in RNNs developed to model long-range dependencies in **sequence data in a more efficient way** than previous models, such as LSTM units.
GRUs have two main gate types.

- Reset Gate
- Update Gate

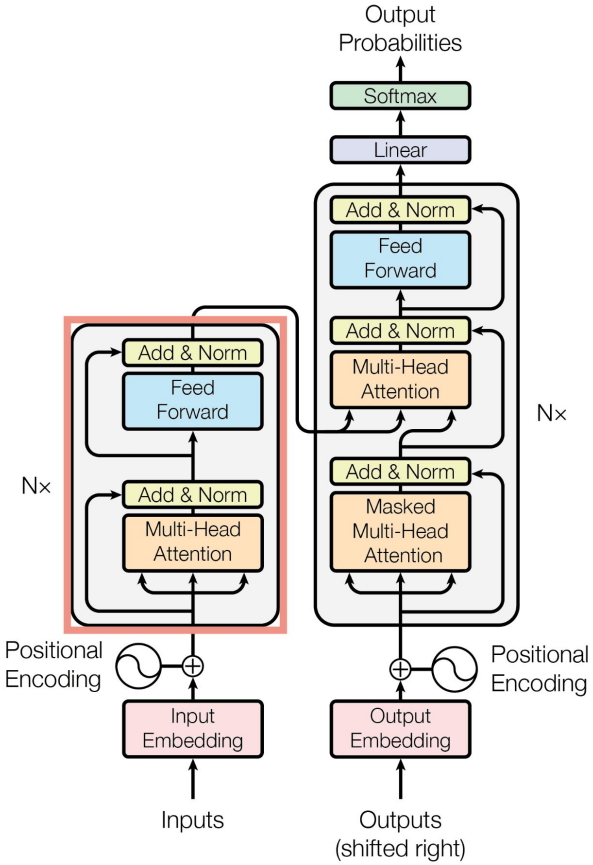


bangkit

Sneak Peek Transformers 👀

Transformers are a type of deep learning model used for natural language processing (NLP) and computer vision (CV) tasks. They utilize a mechanism called **“self-attention”** to process sequential input data.

Transformers can process the entire input data at once, capturing context and relevance.



bangkit

Sequence Prediction

Sequence prediction is a type of machine learning task that involves **predicting the next value or sequence** of values in a given series.

Getting Started with Time Series

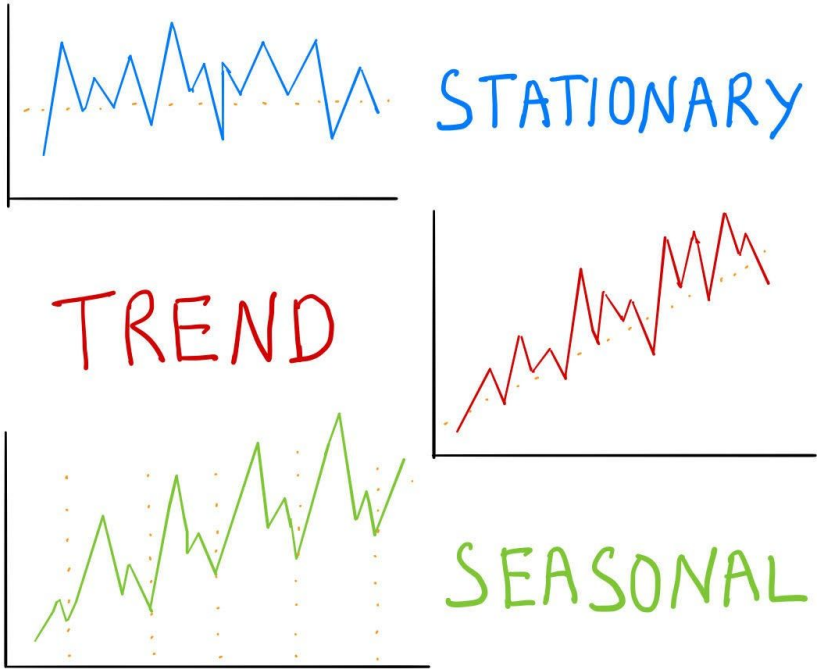
bangkit

What is Time Series?

An **ordered sequence of values** that are usually equally spaced **over time**.

What are they?

- Stock prices
- Weather forecast
- Historical trends



Univariate & Multivariate Time Series

Univariate

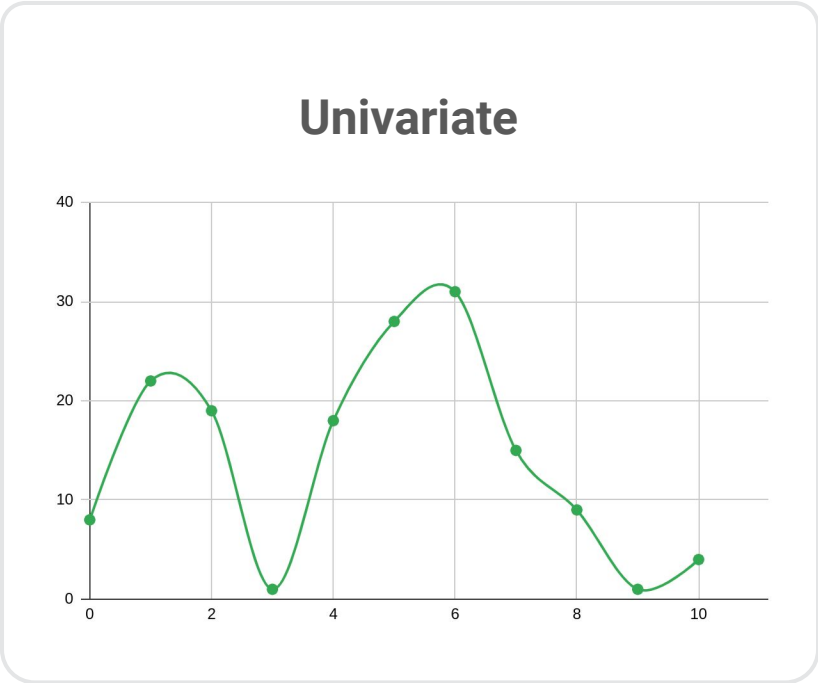
Time	Temperature
05.00 WIB	15 °C
06.00 WIB	15 °C
07.00 WIB	14 °C
08.00 WIB	14 °C
09.00 WIB	16 °C
10.00 WIB	17 °C
11.00 WIB	18 °C
12.00 WIB	19 °C
13.00 WIB	19 °C
14.00 WIB	21 °C
15.00 WIB	22 °C
16.00 WIB	22 °C
17.00 WIB	22 °C
18.00 WIB	21 °C
19.00 WIB	20 °C
20.00 WIB	18 °C
21.00 WIB	18 °C

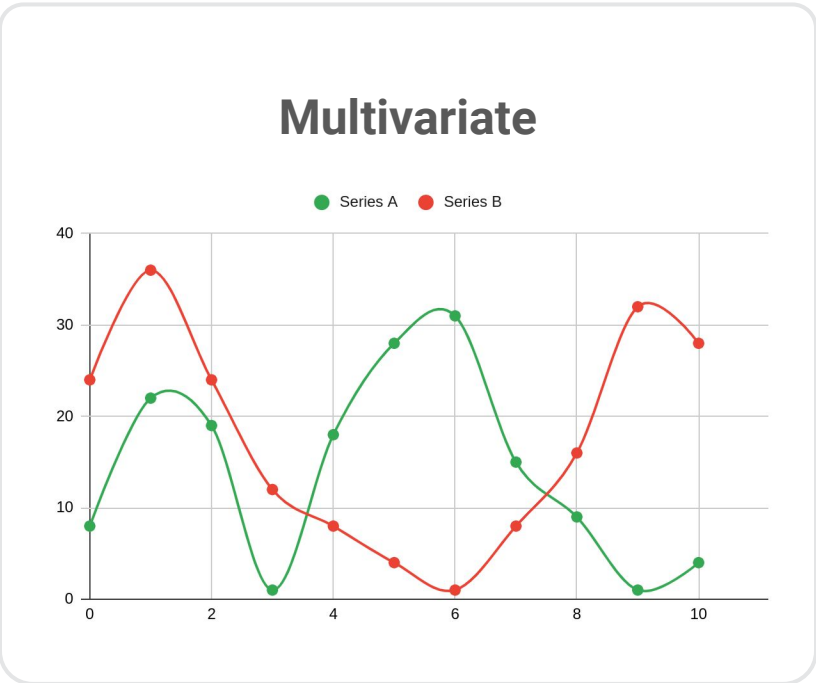
Multivariate

Time	Temperature	Cloud Cover	Dew Point	Humidity	Wind
5:00 am	15 °C	97%	10 °C	74%	8 mph SSE
6:00 am	15 °C	89%	10 °C	75%	8 mph SSE
7:00 am	14 °C	79%	10 °C	76%	7 mph SSE
8:00 am	14 °C	74%	10 °C	77%	7 mph S
9:00 am	16 °C	74%	10 °C	74%	7 mph S
10:00 am	17 °C	74%	11 °C	70%	8 mph S
11:00 am	18 °C	76%	11 °C	65%	8 mph SSW
12:00 am	19 °C	80%	11 °C	60%	8 mph SSW
1:00 pm	19 °C	78%	11 °C	58%	10 mph SW
2:00 pm	21 °C	71%	11 °C	54%	10 mph SW
3:00 pm	22 °C	75%	11 °C	52%	11 mph SW
4:00 pm	22 °C	78%	11 °C	52%	11 mph SW
5:00 pm	22 °C	78%	11 °C	52%	12 mph SW
6:00 pm	21 °C	78%	11 °C	54%	11 mph SW
7:00 pm	20 °C	87%	11 °C	60%	12 mph SW
8:00 pm	18 °C	100%	12 °C	66%	11 mph SSW
9:00 pm	18 °C	100%	12 °C	72%	13 mph SSW

Resource Image: https://github.com/lmoroney/dlaicourse/blob/master/TensorFlow%20In%20Practice/Course%204%20-%20S%2BP/S%2BP_Week_1_Lesson_2.ipynb

Univariate & Multivariate Time Series

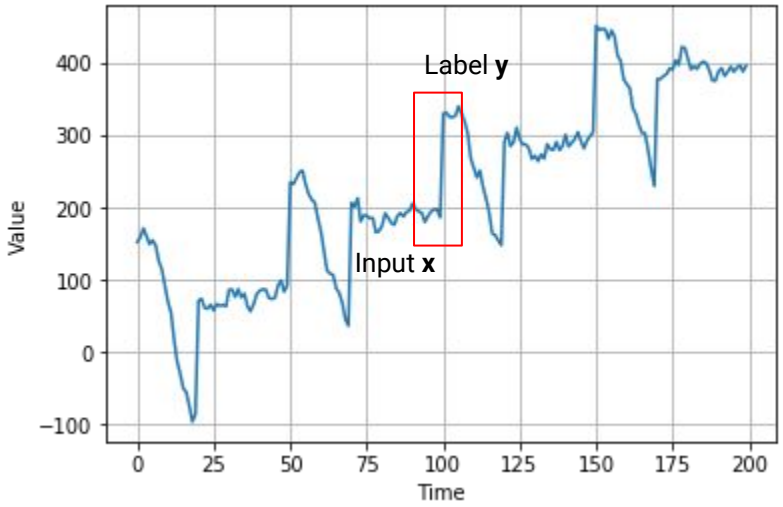




bangkit

Preparing Features and Labels

Windowing the data to create the **features** and **labels**



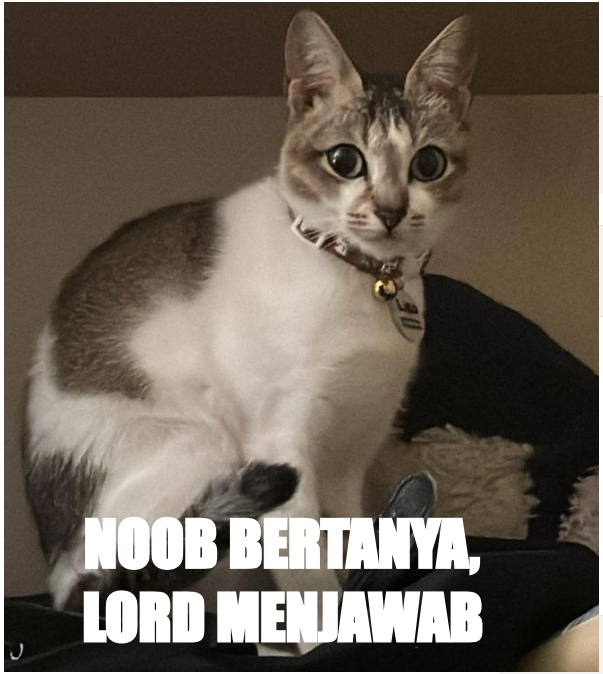
bangkit

Resource Image: https://github.com/lmoroney/dlaicourse/blob/master/TensorFlow%20In%20Practice/Course%204%20-%20S%2BP/S%2BP_Week_1_Lesson_2.ipynb

[Hands-on] Sailing through Time: A Navigator's Guide to Time Series

bangkit

Discussions



Quiz and Feedback

bangkit

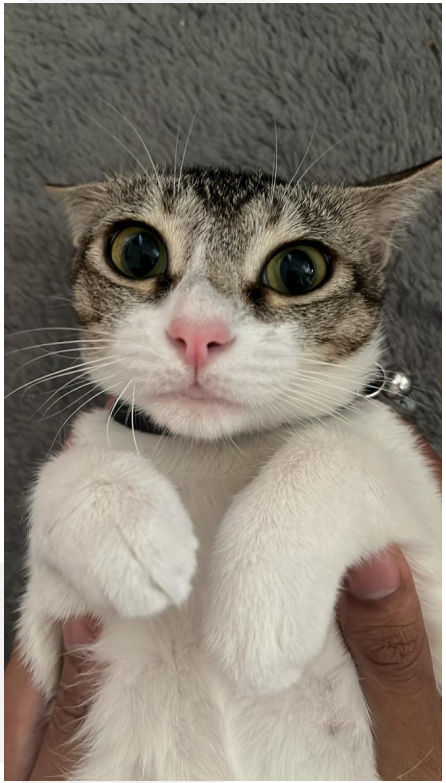
Conclusion

You have learned basic concepts in **NLP and time series prediction**. You also learn about advance deep learning model such as **CNN, RNN, and LSTM**.

These knowledge will give you a solid fundamental concept to becoming a **machine learning practitioner**.

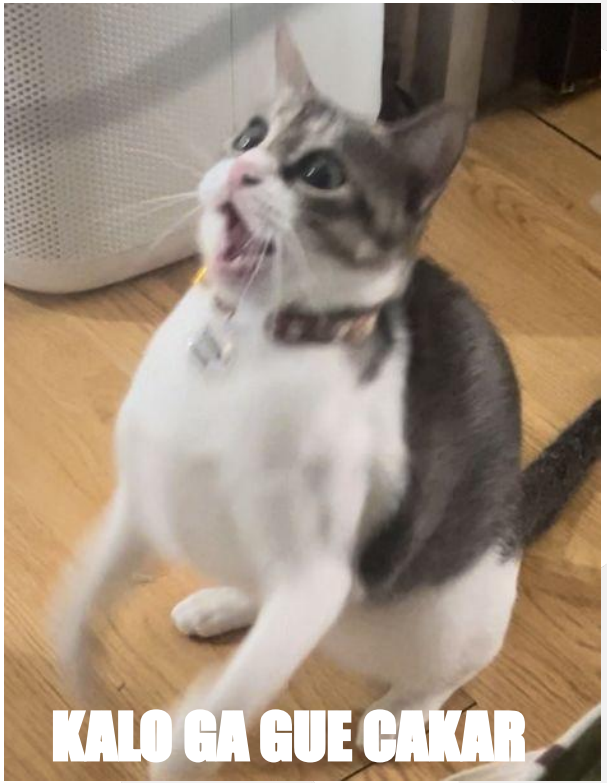
bangkit

Thank You



Follow majikan saia:
Tiktok: @jenniethecatwoman

Currently at ~1500 Followers



bangkit