Introduction to NLP

Data Preparation, Preprocessing, & Modeling

Samuel Cahyawijaya

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

CAIRE
Center for Artificial Intelligence Research

Who Am I

2013 Integrated Modular Avionic Research

2013 Founder of Pelita Cakrawala Inspirasi

2015 Founder of Devarta Kencana Indonesia

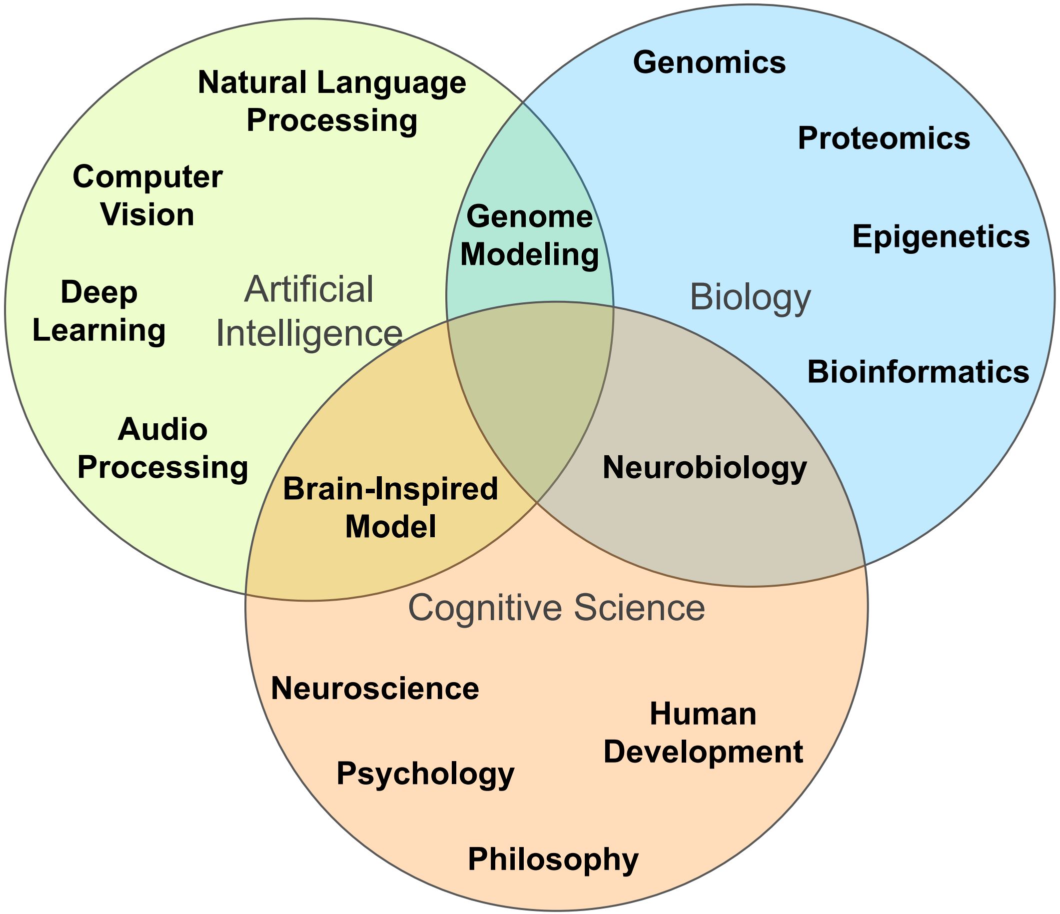
2017 Data Scientist of UangTeman

2018 Research Engineer of Prosa.ai

2019 Senior Data Scientist of JULO

Now Postgraduate Student at ECE HKUST

Area of Interest

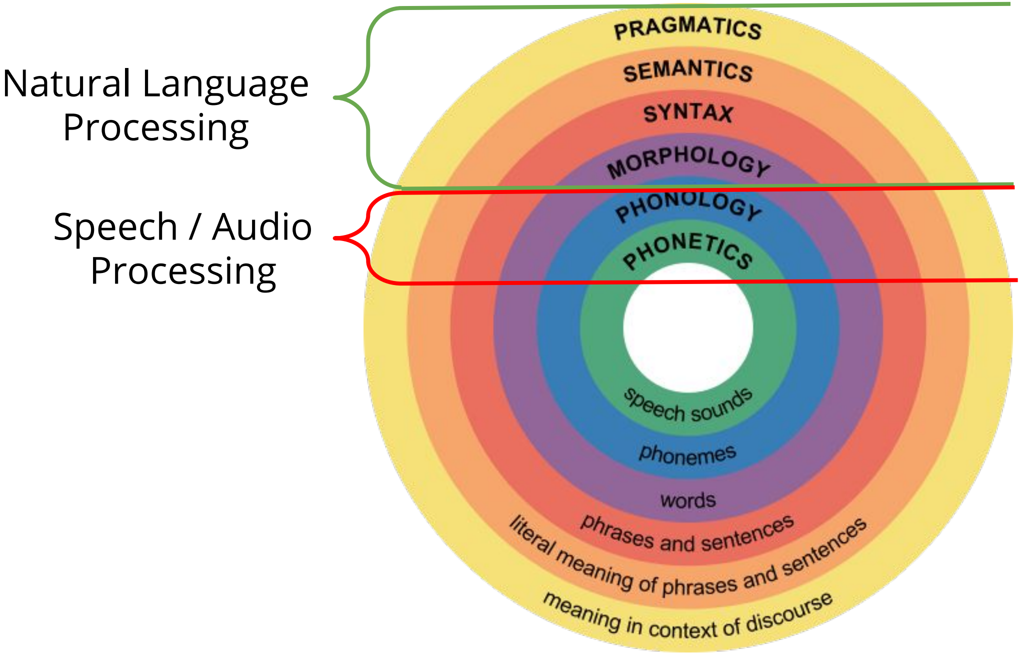


Outline

1. **Basic Concepts of NLP**
2. Preprocessing
3. Modeling
4. Current State of NLP Research
5. Hands-on: Sentiment Analysis
6. Applying Better Learning Strategies

Basic Concepts of NLP

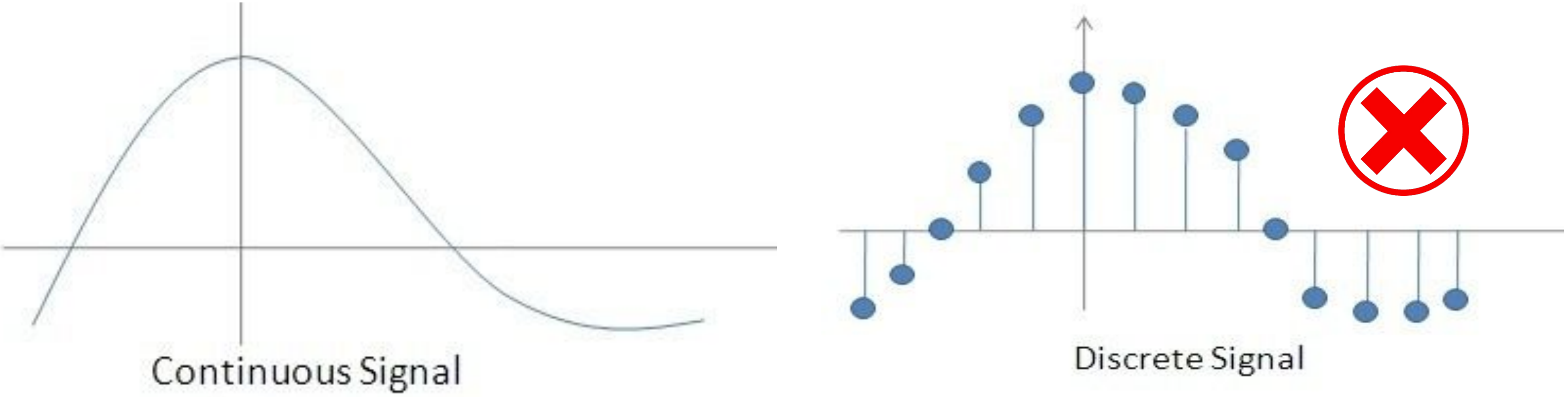
- **Goals**
  Understanding human languages and applying them to enable a more robust human–computer interaction



Basic Concepts of NLP

- **Goals**
  Understanding human languages and applying them to enable a more robust human–computer interaction

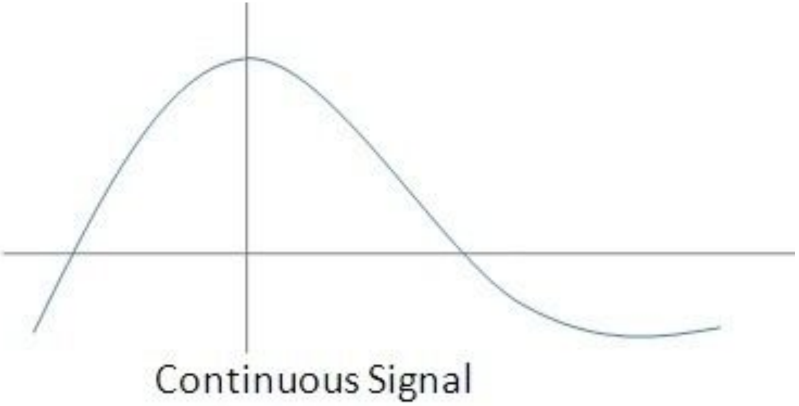
- Any unit of language can be mapped into a **discrete space**
  - Can be character, subword, word, phrase, sentence, etc
  - It is not sampled from a **continuous signal**



Basic Concepts of NLP

- **Goals**
  Understanding human languages and applying them to enable a more robust human–computer interaction

- Any unit of language can be mapped into a **discrete space**
  - Can be character, subword, word, phrase, sentence, etc
  - It is not sampled from a **continuous signal**



Basic Concepts of NLP

- **Goals**
  Understanding human languages and applying them to enable a more robust human–computer interaction
- Any unit of language can be mapped into a **discrete space**
  - Can be character, subword, word, phrase, sentence, etc
  - It is not sampled from a **continuous signal**
- Every language has rules to organize the higher level structure → **Syntax**
  - **Order** is important!!!
  - Because of this NLP data is commonly constructed in form of a **sequence**
  - What is sequence?

“This is a sequence!! This is also a sequence!!!”

  - What is it like to have a **non-sequential language**?

ARRIVAL

(2016)

How to Prepare Data in NLP?

- Collect the **sequence**
  - Sentence
  - Paragraph
  - Document
- Collect the label of the sequence
  - Sequence classification (sentiment analysis, spam filtering, etc)
  - Sequence labeling (POS tag, named entity recognition, span extraction, etc)
  - Pair-sequence classification (document similarity, entailment, etc)
  - Translation
  - Abstractive summarization
  - etc

Preprocessing

“Balonku ada lims rupa-rupa warnany”

↓ Transformation

“balonku ada lima rupa-rupa warnanya”

↓ Tokenization

[balon, ku, ada, lima, rupa, -, rupa, warna, nya]

↓ Word2Index

[0, 5, 1, 3, 2, 5, 2, 4, 5]

↓ Encoding

Modeling representations

Vocabulary

balon	0
ada	1
rupa	2
lima	3
warna	4
UNK	5

Language Structure

- **Grapheme** : “A”, “B”, “C”, “?”, “!”, “$”, “~”, etc
- **Morphology** : The smallest units with meaning, it is **not necessarily a word**
    - **English** : untouchable → “un”, “touch”, “able”
    - **Indonesian** : memelihara → “mem”, “pelihara”
- **Syntax** : Budi memakan nasi  memakan Budi nasi 
- **Semantic** : Understanding the meaning of word
- **Pragmatic** : Understanding the meaning of word in the underlying context

    <u>Budi</u> menyantap <u>internet</u> di warung kopi

    <u>Budi</u> baikmu akan selalu kukenang

    <u>Internet</u> memegang peranan penting bagi pertumbuhan ekonomi

Why is language structure important?

- Representation of the **token** (smallest unit of a sequence)
- Let say we have a pretty simple language called “**BABABA”** consisting of only **A** and **B** characters. From these 2 characters, we construct the following rules:
    - The language consists of 100 **base words**: **“BA”**, **“BABA”**, **“BABABA”**, **...**
    - The language consists of 10 **prefixes**: **“AB”** , **“ABAB”**, **“ABABAB”**, ...
    - The language consists of 10 **suffices**: **“ABB”** , **“ABBABB”**, **“ABBABBABB”**, ...
    - Any word can have any prefix and suffix
    - Any combination of two words can construct a **phrase**
- Let’s define our **tokenization** approach
    - If we consider token to be **grapheme / character**, our **vocabulary** size will be only **2**
    - If we consider token to be **morpheme**, our **vocabulary** size will be **120**
    - If we consider token to be **word**, our **vocabulary** size will be **10,000**
    - If we consider token to be **phrase**, our **vocabulary** size will be **100,000,000**

Why is language structure important?

- Representation of the **token** (smallest unit of a sequence)
- Let say we have a pretty simple language called "**BABABA"** consisting of only **A** and **B** characters. From this 2 characters, we construct several concepts as follow
  - The language consists of 100 **base words**: **"BA"**, **"BABA"**, **"BABABA"**, **...**
  - The language consists of 10 **prefixes**: **"AB"** , **"ABAB"**, **"ABABAB"**, ...
  - The language consists of 10 **suffices**: **"ABB"** , **"ABBABB"**, **"ABBABBABB"**, ...
  - Any word can have any prefix and suffix
  - Any combination of two words can construct a **phrase**

Vocabulary Size
Increase dramatically

- Let's define our **tokenization** approach
  - If we consider token to be **grapheme / character**, our **vocabulary** size will be only **2**
  - If we consider token to be **morpheme**, our **vocabulary** size will be **120**
  - If we consider token to be **word**, our **vocabulary** size will be **10,000**
  - If we consider token to be **phrase**, our **vocabulary** size will be **100,000,000**

Handling Vocabulary Size

- What is the problem?
  - Larger vocabulary means more model parameters
  - The occurences of each token can be very skewed → some tokens are barely learnt
  - If token of the vocabulary is too low-level, it is hard for model to learn **higher semantic**
- How do we reduce vocabulary size?
  - Limit number of vocab (uncovered token will be replaced as **unknown token**)
  - Stemming & Lemmatization
  - Word normalization
  - Stop word removal
  - Standardize case
- How to increase vocabulary size?
  - n-gram → another representation of token made by combining nearby tokens
  - e.g: “aku suka makan pisang” → [“aku suka”, “suka makan”, “makan pisang”]

Some details on Preprocessing

Text Normalization

Raw	Normalized
2moro 2mrrw 2morrow 2mrw tomrw	tomorrow
b4	before
otw	on the way
:) :-) ;-)	smile

Stemming & Lemmatization

Stemming	Lemmatization
adjustable → adjust	was → (to) be
formality → formaliti	better → good
formaliti → formal	meeting → meeting
airliner → airlin ⚠	

Stopwords

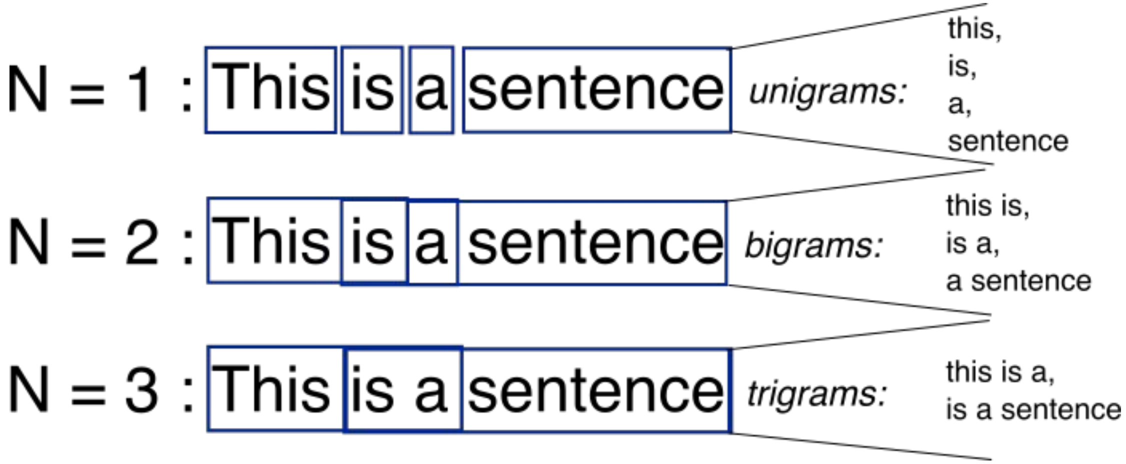
758 lines (758 sloc) | 6.29 KB

```
ada
adalah
adanya
adapun
agak
agaknya
agar
```

Some details on Preprocessing

n-gram

Can greatly increase the vocabulary size!!!



Out of Vocabulary (OOV)

Let’s define our **tokenization** approach

- If we consider token to be **grapheme / character**, our **vocabulary** size will be only **2**
- If we consider token to be **morpheme**, our **vocabulary** size will be **120**
- If we consider token to be **word**, our **vocabulary** size will be **10,000**
- If we consider token to be **phrase**, our **vocabulary** size will be **100,000,000**

- In real case, given a training corpus for each language we might be able to list most of the **graphemes**
- But, can we capture most of the possible **words**?
- What about **phrases**, **n-gram**, or even **sentences**? Can we capture all combinations of them to cover all of the possibilities?

Modeling Representation

Vocabulary

balon	0
ada	1
rupa	2
lima	3
warna	4
UNK	5

“Balonku ada lims rupa-rupa warnany”

Transformation (normalization,)

“balonku ada lima rupa-rupa warnanya”

Tokenization

[balon, ku, ada, lima, rupa, -, rupa, warna, nya]

Word2Index

[0, 5, 1, 3, 2, 5, 2, 4, 5]

Encoding

Modeling representations

- Bag-of-words
- One-hot-encoding
- Dense vector

Bag-of-words

Vocabulary

balon	0
ada	1
rupa	2
lima	3
warna	4
UNK	5

"Balonku ada lims rupa-rupa warnany"

Transformation (normalization,)

"balonku ada lima rupa-rupa warnanya"

Tokenization

[balon, ku, ada, lima, rupa, -, rupa, warna, nya]

Word2Index

[0, 5, 1, 3, 2, 5, 2, 4, 5]

Encoding

[1, 1, 2, 1, 1, 2]

Bag-of-words

Vocabulary

balon	0
ada	1
rupa	2
lima	3
warna	4
UNK	5

“Balonku ada lims rupa-rupa warnany”

↓ Transformation (normalization,)

“balonku ada lima rupa-rupa warnanya”

↓ Tokenization

[balon, ku, ada, lima, rupa, -, rupa, warna, nya]

↓ Word2Index

[0, 5, 1, 3, 2, 5, 2, 4, 5]

↓ Encoding

[1, 1, 2, 1, 1, 2]

This is the representation for the whole sentence!!
Missing sequence information!!

One-hot-encoding

Vocabulary

balon	0
ada	1
rupa	2
lima	3
warna	4
UNK	5

"Balonku ada lims rupa-rupa warnany"

↓ Transformation (normalization,)

"balonku ada lima rupa-rupa warnanya"

↓ Tokenization

[balon, ku, ada, lima, rupa, -, rupa, warna, nya]

↓ Word2Index

[0, 5, 1, 3, 2, 5, 2, 4, 5]

↓ Encoding

[[1, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 1],
[0, 1, 0, 0, 0, 0],
...,
[0, 0, 0, 0, 0, 1]]

One-hot-encoding

Vocabulary

balon	0
ada	1
rupa	2
lima	3
warna	4
UNK	5

“Balonku ada lims rupa-rupa warnany”

Transformation (normalization,)

“balonku ada lima rupa-rupa warnanya”

Tokenization

[balon, ku, ada, lima, rupa, -, rupa, warna, nya]

Word2Index

[0, 5, 1, 3, 2, 5, 2, 4, 5]

Encoding

[[1, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 1],
[0, 1, 0, 0, 0, 0],
...,
[0, 0, 0, 0, 0, 1]]

Sparse and Inefficient

Dense Vector

“Balonku ada lims rupa-rupa warnany”

Transformation (normalization,)

“balonku ada lima rupa-rupa warnanya”

Tokenization

[balon, ku, ada, lima, rupa, -, rupa, warna, nya]

Word2Index

[0, 5, 1, 3, 2, 5, 2, 4, 5]

Encoding

[[0.2, 5.0, 1.3],
[9.9, 8.8, 7.7],
[3.3, 1.0, 0.3],
...,
[9.9, 8.8, 7.7]]

Vocabulary

balon	0
ada	1
rupa	2
lima	3
warna	4
UNK	5

Dense Map

0	[0.2, 5.0, 1.3]
1	[3.3, 1.0, 0.3]
2	[0.2, 3.8, 3.7]
3	[1.1, 0.1, 1.2]
4	[5.2, 3.0, 1.2]
5	[9.9, 8.8, 7.7]

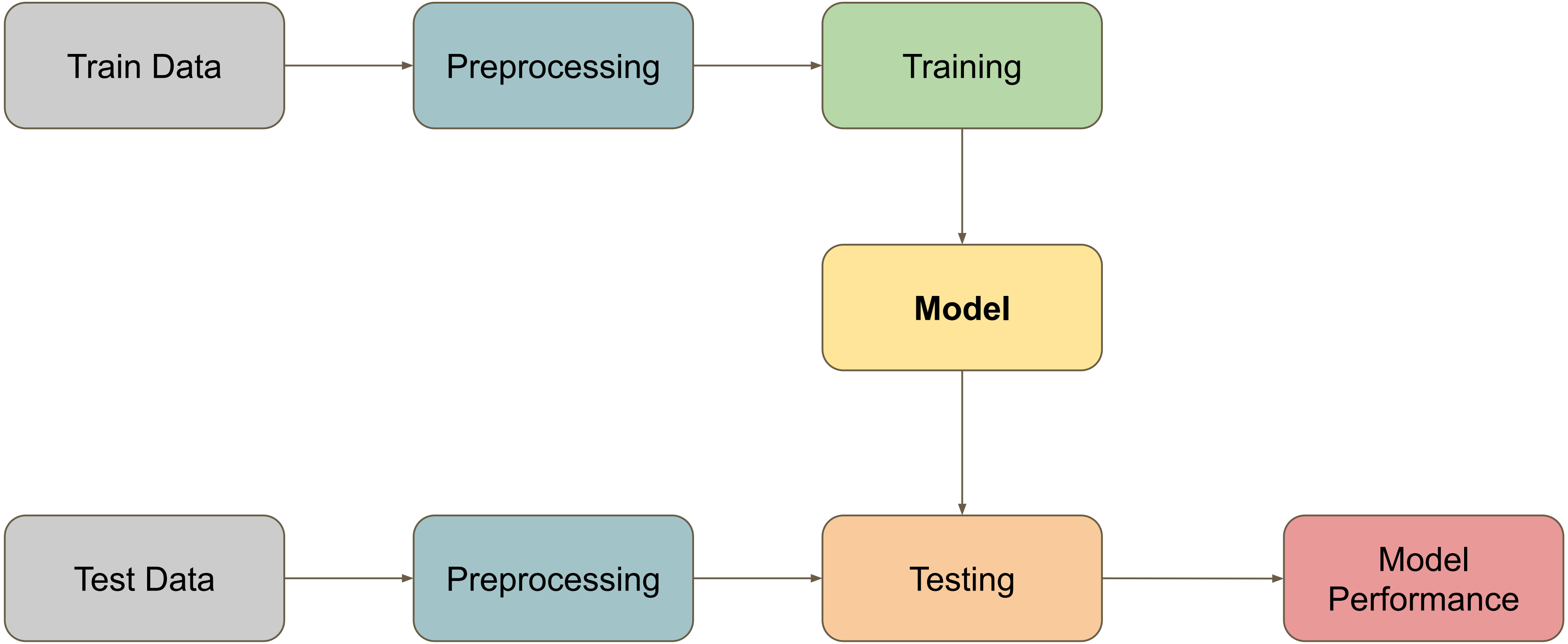
Why Preprocessing is Important?

- Controlling the number of vocabulary
- Generating vocabulary that are meaningful and can be well-trained
  - Has more uniform token's occurrences
  - Can cover most of the tokens in any possible sequence
- Generate better vocabulary with minimal information loss
  - Loss of context information
  - Loss of sequence information
  - Distorted meaning

The current trend of NLP preprocessing

- **(Optional)** case standardization
- **Subword** level tokenization (**SentencePiece & BPE**)
- Considering **<space>** character
- Use **unknown token** to replace missing subwords
- Use **dense vector** representation as the input

Let’s go for modeling



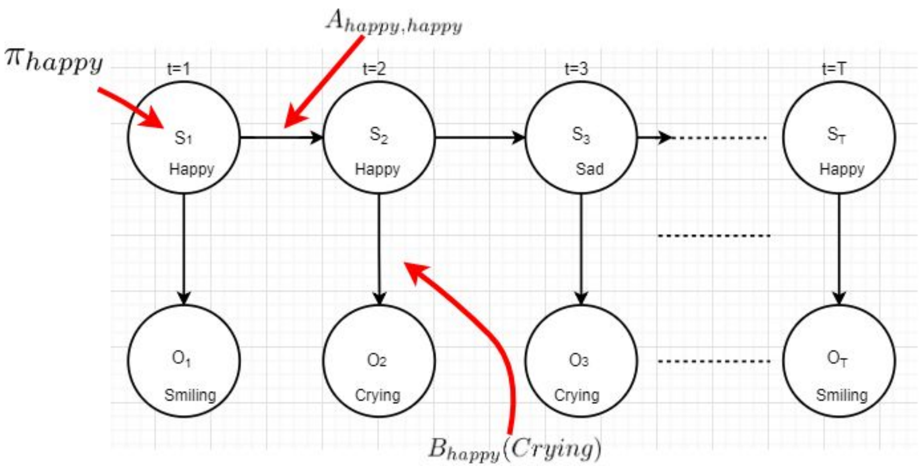
The history of NLP models

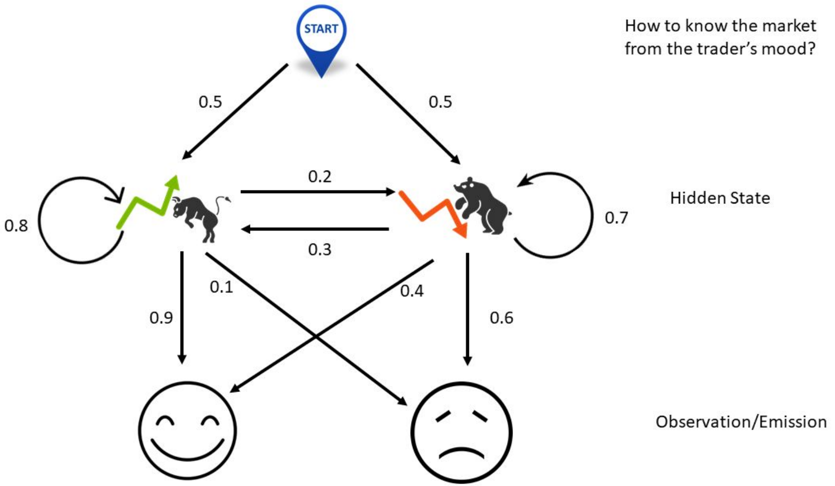
- Bag-of-word-based model (1954) → kNN, Bayesian, Tree, Forest, SVM, etc
    - Problem? **No sequence information!!**
        - aku mau kamu pergi bersama dia
        - aku mau dia pergi bersama kamu
        - kamu mau aku pergi bersama dia
        - kamu mau dia pergi bersama aku
        - dia mau aku pergi bersama kamu
        - dia mau kamu pergi bersama aku

aku	mau	pergi	kamu	bersama	dia
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1

The history of NLP models

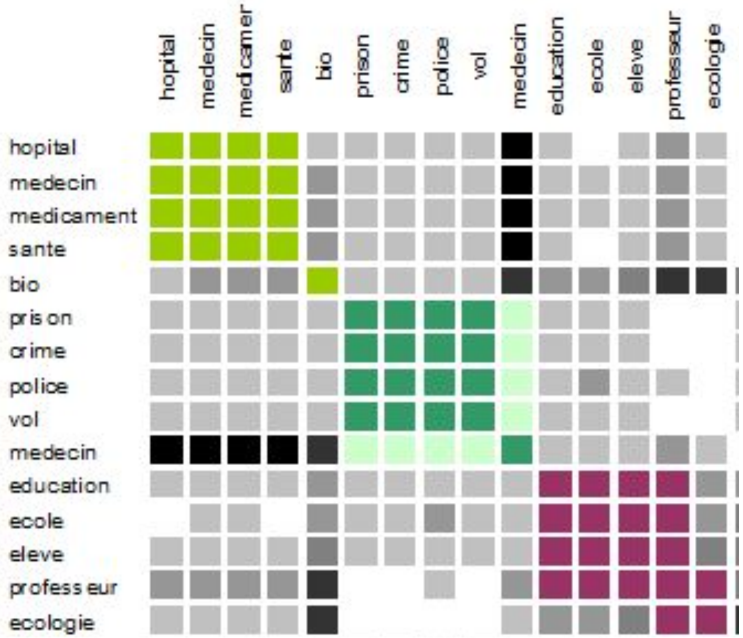
- Bag-of-word based model (1954) → kNN, Bayesian, Tree, Forest, SVM, etc
- Bayesian-network based model (1960) → HMM and DBN
    - Problem? **Markovian assumption, one-hot-encoding is inefficient**





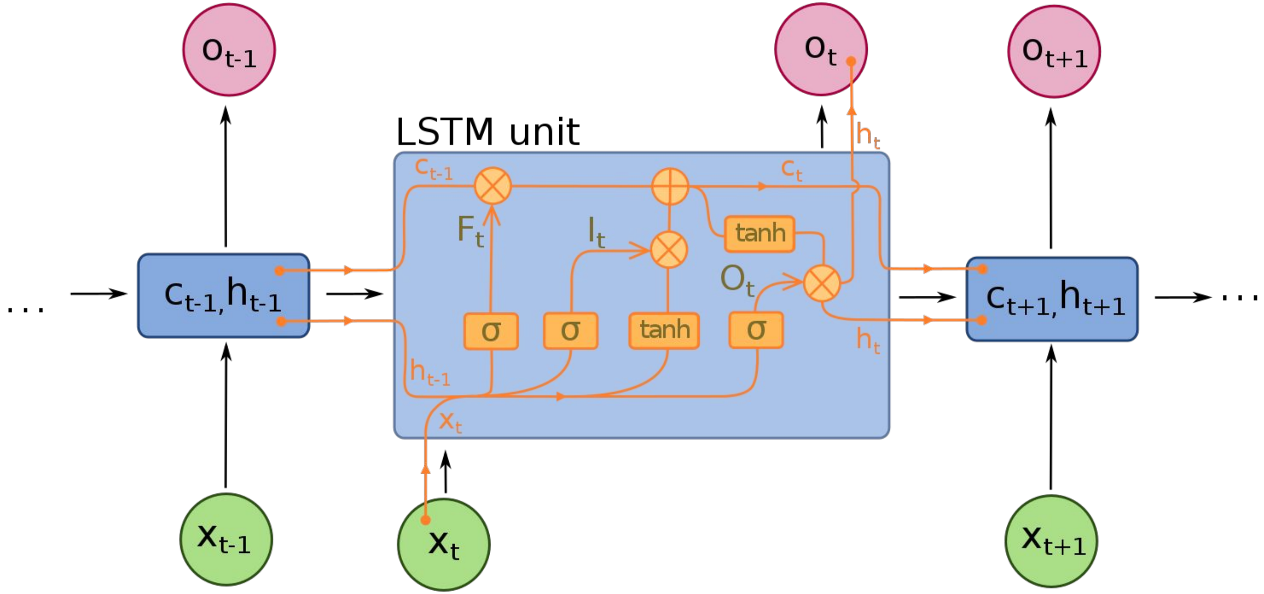
The history of NLP models

- Bag-of-word based model (1954) → kNN, Bayesian, Tree, Forest, SVM, etc
- Bayesian-network based model (1960) → HMM and DBN
- Latent Semantic Analysis (1980) → Commonly use for topic modeling



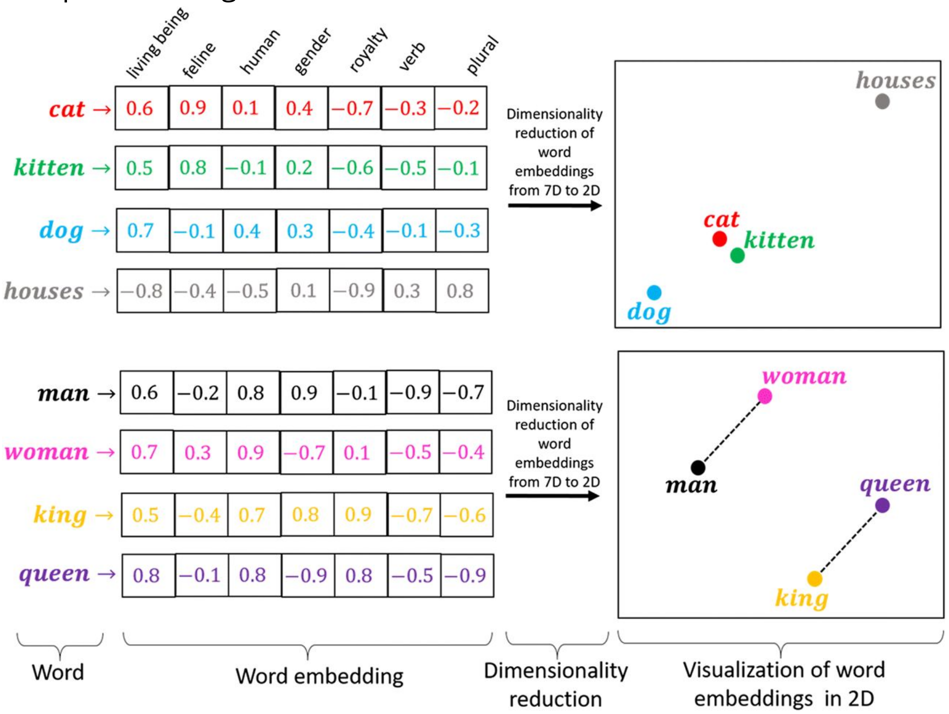
The history of NLP models

- Bag-of-word based model (1954) → kNN, Bayesian, Tree, Forest, SVM, etc
- Bayesian-network based model (1960) → HMM and DBN
- Latent Semantic Analysis (1980) → Commonly use for topic modeling
- LSTM (1997) → Well-known RNN-based model



The history of NLP models

- Bag-of-word based model (1954) → kNN, Bayesian, Tree, Forest, SVM, etc
- Bayesian-network based model (1960) → HMM and DBN
- Latent Semantic Analysis (1980) → Commonly use for topic modeling
- LSTM (1997) → Well-known RNN-based model
- **Word Embedding (2000)**



The history of NLP models

- Bag-of-word based model (1954) → kNN, Bayesian, Tree, Forest, SVM, etc
- Bayesian-network based model (1960) → HMM and DBN
- Latent Semantic Analysis (1980) → Commonly use for topic modeling
- LSTM (1997) → Well-known RNN-based model
- Word Embedding (2000)
- Word2vec (2013) → Faster training for word embedding **(Google)**
- GRU (2014) → Another well-known RNN-based model
- FastText (2015) → Subword based word embedding **(Facebook)**

The history of NLP models

- Bag-of-word based model (1954) → kNN, Bayesian, Tree, Forest, SVM, etc
- Bayesian-network based model (1960) → HMM and DBN
- Latent Semantic Analysis (1980) → Commonly use for topic modeling
- Word Embedding (2000)
- Word2vec (2013) → Faster training for word embedding **(Google)**
- GRU (2014) → Another well-known RNN-based model
- FastText (2015) → Subword based word embedding **(Facebook)**
- Transformer (2017) → Attention-based sequence processor **(Google)**
  - Transformer can process sequence in parallel → **Faster training and inference time**
  - A token in Transformer model can attend to any token in the sequence (**No markovian assumption**)

The history of NLP models

- Bag-of-word based model (1954) → kNN, Bayesian, Tree, Forest, SVM, etc
- Bayesian-network based model (1960) → HMM and DBN
- Latent Semantic Analysis (1980) → Commonly use for topic modeling
- Word Embedding (2000)
- Word2vec (2013) → Faster training for word embedding **(Google)**
- GRU (2014) → Another well-known RNN-based model
- FastText (2015) → Subword based word embedding **(Facebook)**
- Transformer (2017) → Attention-based sequence processor **(Google)**
- ELMo (2018) → First contextualized embedding model
    - Bidirectional LSTM based model
    - Trained with **1B words** benchmark corpus
    - Developed by: **Allennlp**

The history of NLP models

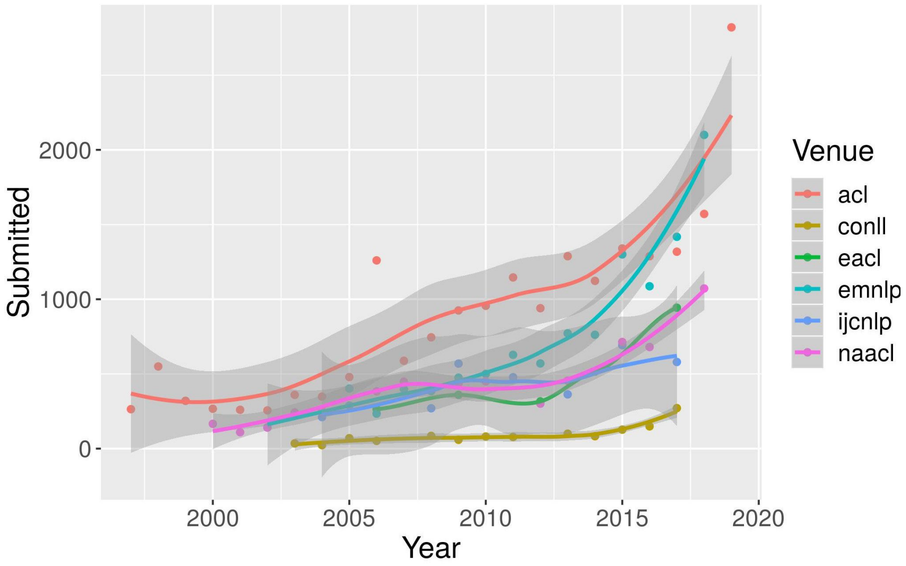
- Bag-of-word based model (1954) → kNN, Bayesian, Tree, Forest, SVM, etc
- Bayesian-network based model (1960) → HMM and DBN
- Latent Semantic Analysis (1980) → Commonly use for topic modeling
- Word Embedding (2000)
- Word2vec (2013) → Faster training for word embedding **(Google)**
- GRU (2014) → Another well-known RNN-based model
- FastText (2015) → Subword based word embedding **(Facebook)**
- Transformer (2017) → Attention-based sequence processor **(Google)**
- ELMo (2018) → First contextualized embedding model (Bidirectional)
- BERT (2019) → Transformer based contextualized embedding model (Google)
  - Transformer-encoder only models
  - Pre-trained on **Wikipedia (2,500M words)** and **Book Corpus (800M words) [~30GB]**
- GPT-2 (2019) → Transformer based language generation model (OpenAI)
  - Transformer-decoder only models
  - Pre-trained on **8M web pages (~40GB)**

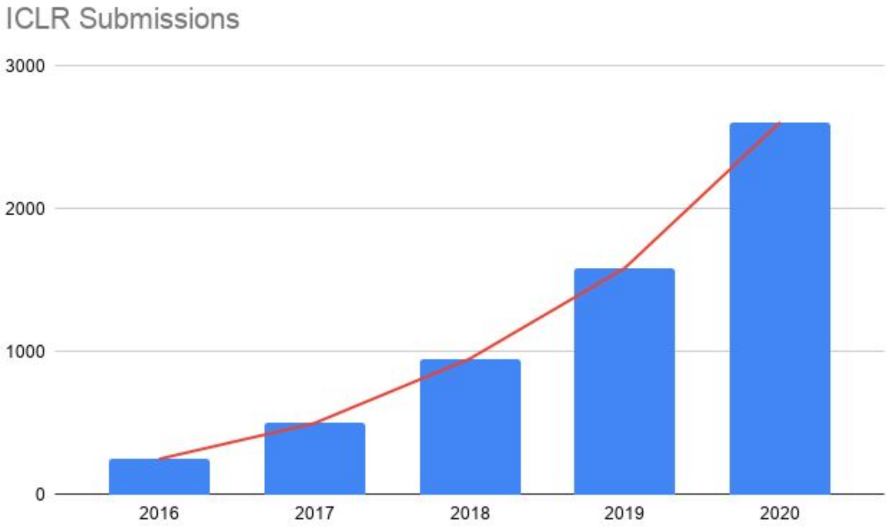
Beyond BERT & GPT-2

- Encoder-only models
  - BERT → Transformer based contextualized embedding model (Google)
  - RoBERTa → Robustly trained BERT (Facebook)
  - ALBERT → **Factorized** BERT (Google)
  - DistilBERT → Smaller BERT model trained from **Distillation**
  - MBERT → BERT model trained on **multilingual** data (Google)
  - XLM → Similar to MBERT but different **tokenization** and **pre-training** (Facebook)
  - XLM-R → Similar to XLM but with **RoBERTa** model and larger training corpus (Facebook)
- Decoder-only models
  - GPT-2 → Transformer based language generation model (Open AI)
  - UniLM → Transformer based language generation model (Microsoft
  - XNLG → Multilingual model for language generation (Microsoft)
  - GPT-3 → Extremely large language generation model (OpenAI & Microsoft)
- Encoder-decoder models
  - T5 → Encoder-decoder based transformer model (Google)
  - BART → Encoder-decoder based transformer model (Facebook)
  - MASS → Encoder-decoder based transformer model (Microsoft)

And many more….

Why so many models in recent years?





IndoBERT from IndoNLU

BERT model for Indonesian Natural Language Understanding

IndoBERT Models

IndoBERT

Base

124.5M parameters

Large

335.1M parameters

IndoBERT-lite

Base

11.7M parameters

Large

17.7M parameters

Indo4B Dataset

23GB+ of Indonesian data

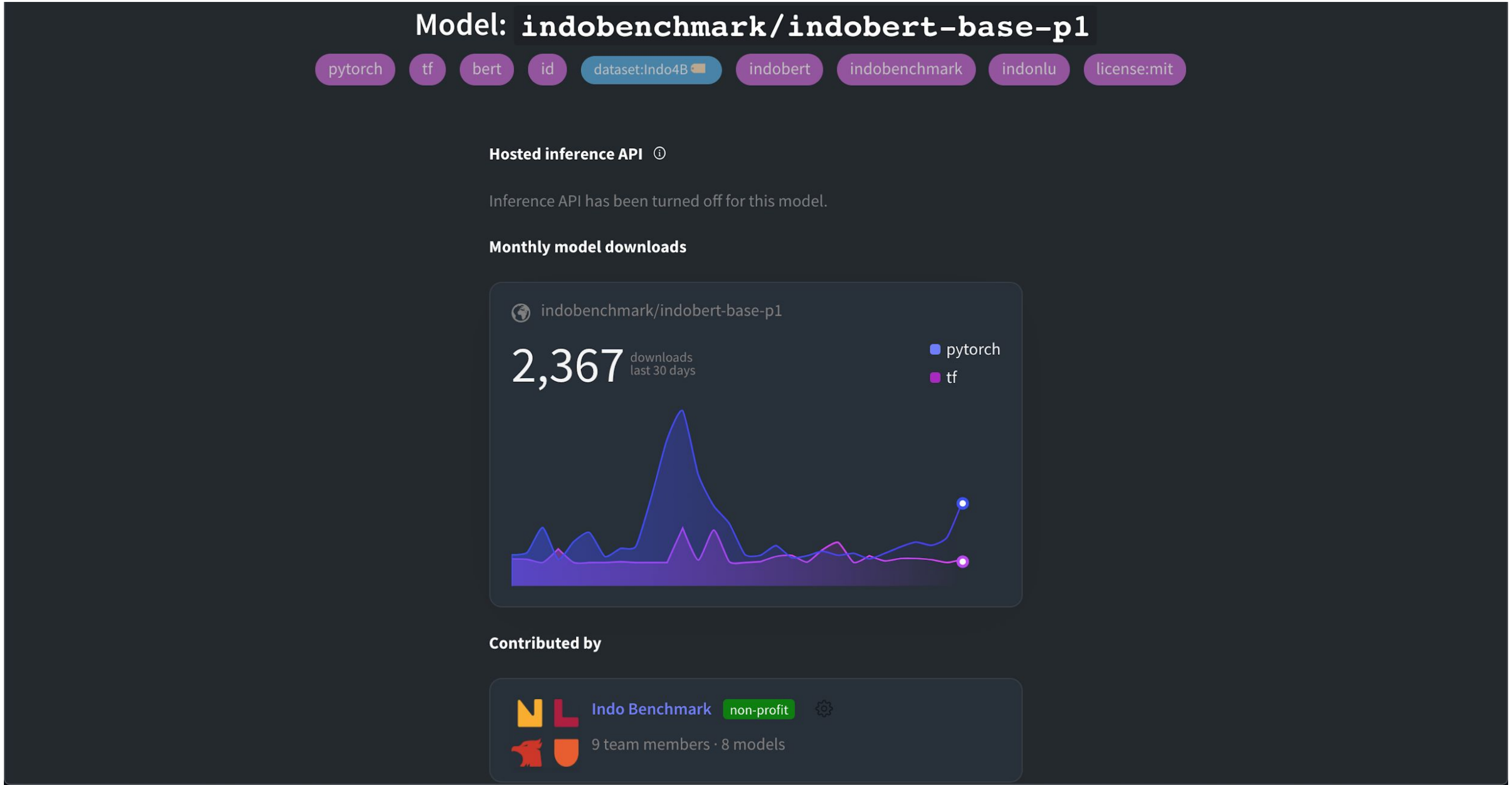
3.5+ billion words

15 sources

Colloquial and Formal

Our models are hosted in HuggingFace!

https://huggingface.co/indobenchmark



Our models are hosted in HuggingFace!

```
from transformers import BertTokenizer, AutoModel
tokenizer = BertTokenizer.from_pretrained("indobenchmark/indobert-base-p1")
model = AutoModel.from_pretrained("indobenchmark/indobert-base-p1")
```

Models	**Size**	**Phase1**	**Phase2**
$\text{IndoBERT}_{\text{BASE}}$	124.5M	indobenchmark/indobert-base-p1	indobenchmark/indobert-base-p2
$\text{IndoBERT}_{\text{LARGE}}$	335.2M	indobenchmark/indobert-large-p1	indobenchmark/indobert-large-p2
$\text{IndoBERT-lite}_{\text{BASE}}$	11.7M	indobenchmark/indobert-lite-base-p1	indobenchmark/indobert-lite-base-p2
$\text{IndoBERT-lite}_{\text{LARGE}}$	17.7M	indobenchmark/indobert-lite-large-p1	indobenchmark/indobert-lite-large-p2

12 Tasks IndoNLU Benchmark

Dataset	|Train|	|Valid|	|Test|	Task Description	#Label	#Class	Domain	Style
				Single-Sentence Classification Tasks				
EmoT†	3,521	440	442	emotion classification	1	5	tweets	colloquial
SmSA	11,000	1,260	500	sentiment analysis	1	3	general	colloquial
CASA	810	90	180	aspect-based sentiment analysis	6	3	automobile	colloquial
HoASA†	2,283	285	286	aspect-based sentiment analysis	10	4	hotel	colloquial
				Sentence-Pair Classification Tasks				
WReTE†	300	50	100	textual entailment	1	2	wiki	formal
				Single-Sentence Sequence Labeling Tasks				
POSP†	6,720	840	840	part-of-speech tagging	1	26	news	formal
BaPOS	8,000	1,000	1,029	part-of-speech tagging	1	41	news	formal
TermA	3,000	1,000	1,000	span extraction	1	5	hotel	colloquial
KEPS	800	200	247	span extraction	1	3	banking	colloquial
NERGrit†	1,672	209	209	named entity recognition	1	7	wiki	formal
NERP†	6,720	840	840	named entity recognition	1	11	news	formal
				Sentence-Pair Sequence Labeling Tasks				
FacQA	2,495	311	311	span extraction	1	3	news	formal

Table 1: Task statistics and descriptions. †We create new splits for the dataset.

12 Tasks IndoNLU Benchmark

Dataset	|Train|	|Valid|	|Test|	Task Description	#Label	#Class	Domain	Style
Single-Sentence Classification Tasks								
EmoT†	3,521	440	442	emotion classification	1	5	tweets	colloquial
SmSA	11,000	1,260	500	sentiment analysis	1	3	general	colloquial
CASA	810	90	180	aspect-based sentiment analysis	6	3	automobile	colloquial
HoASA†	2,283	285	286	aspect-based sentiment analysis	10	4	hotel	colloquial
Sentence-Pair Classification Tasks								
WReTE†	300	50	100	textual entailment	1	2	wiki	formal
Single-Sentence Sequence Labeling Tasks								
POSP†	6,720	840	840	part-of-speech tagging	1	26	news	formal
BaPOS	8,000	1,000	1,029	part-of-speech tagging	1	41	news	formal
TermA	3,000	1,000	1,000	span extraction	1	5	hotel	colloquial
KEPS	800	200	247	span extraction	1	3	banking	colloquial
NERGrit†	1,672	209	209	named entity recognition	1	7	wiki	formal
NERP†	6,720	840	840	named entity recognition	1	11	news	formal
Sentence-Pair Sequence Labeling Tasks								
FacQA	2,495	311	311	span extraction	1	3	news	formal

Multilabel

Pair Sentence

Classification

Sequence Labeling

Pair Sentence

Table 1: Task statistics and descriptions. †We create new splits for the dataset.

Tutorial

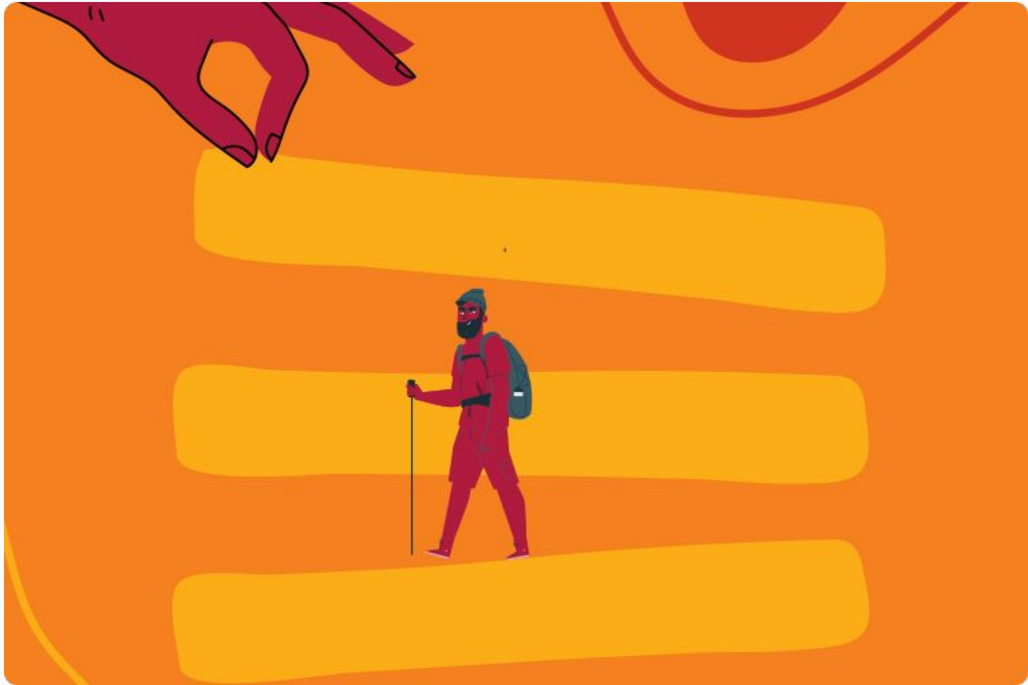
https://github.com/indobenchmark/indonlu

SmSA
Sequence Classification

WreTe
Pair-Sequence Classification

NERGrit
Sequence Labeling

CASA
Multilabel Seq. Classification



Visit our homepage

https://indobenchmark.com

https://github.com/indobenchmark

```
@inproceedings{wilie2020indonlu,
    title={IndoNLU: Benchmark and Resources for
        Evaluating Indonesian Natural Language Understanding},
    author={Bryan Wilie and Karissa Vincentio and Genta Indra Winata
       and Samuel Cahyawijaya and X. Li and Zhi Yuan Lim
       and S. Soleman and R. Mahendra and Pascale Fung
       and Syafri Bahar and A. Purwarianti},
    booktitle={Proceedings of the 1st Conference of the Asia-Pacific Chapter
        of the Association for Computational Linguistics and
        the 10th International Joint Conference on Natural Language Processing},
year={2020}
}
```