NLP Basic

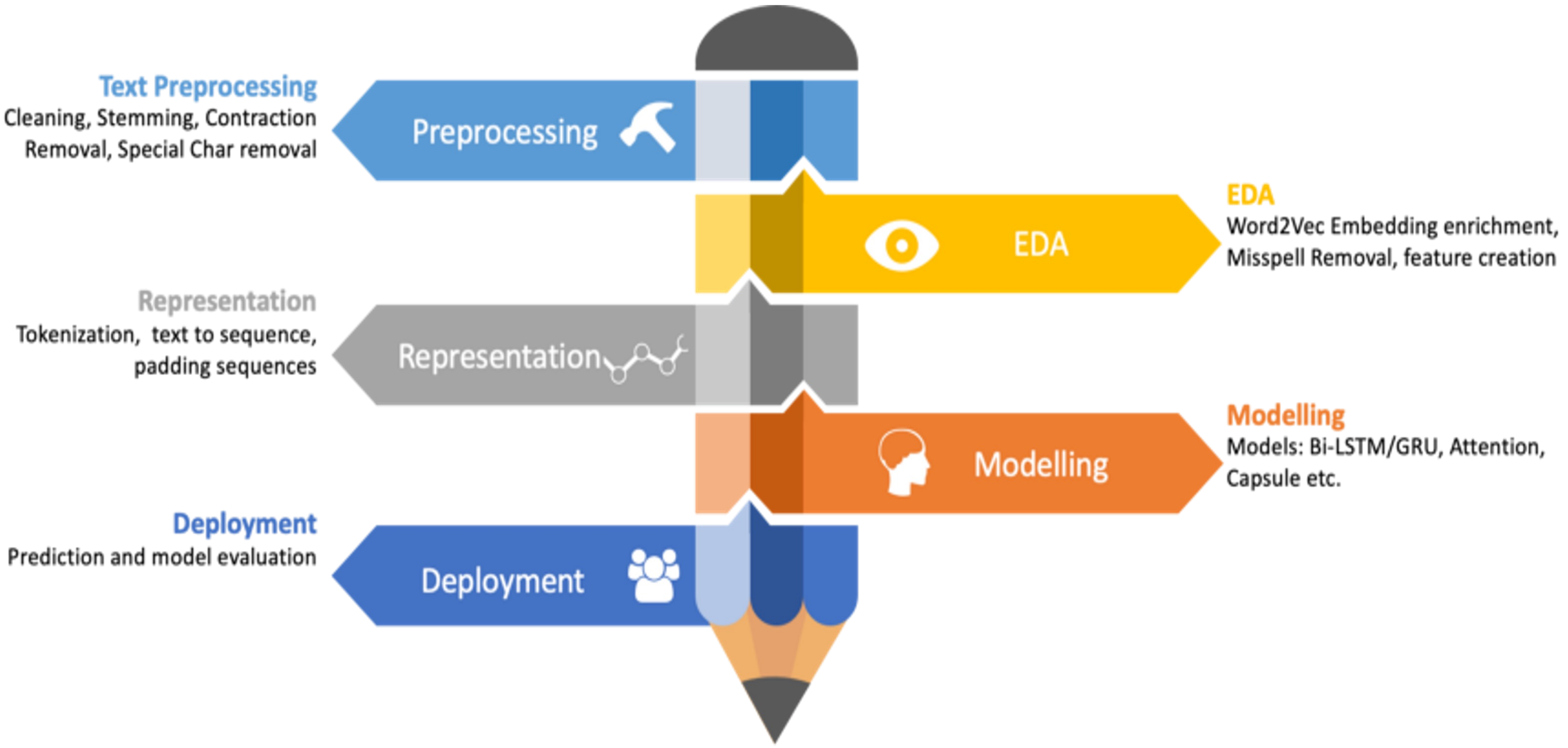
02 – PREPROCESSING TOKENIZATION

AI VIET NAM
Nguyen Quoc Thai

Year 2023

CONTENT

NLP pipeline


Text Preprocessing
Cleaning, Stemming, Contraction Removal, Special Char removal
Preprocessing
EDA
Word2Vec Embedding enrichment, Misspell Removal, feature creation
EDA
Representation
Tokenization, text to sequence, padding sequences
Representation
Modelling
Models: Bi-LSTM/GRU, Attention, Capsule etc.
Modelling
Deployment
Prediction and model evaluation
Deployment

AI VIET NAM
@aivietnam.edu.vn

1 – Text preprocessing

Problems?

- **Source**

  Blogs, Facebook, News,…

- **Most raw text data:**

  include: URLs, HTML tags,…

  short words, making typo errors

- **Example**

```
@AppleSupport causing the reply to be disregarded and the
tapped notification under the keyboard is opened😡😡😡
@82476 🤔 <p> We'd like to help Sam, which number is calin
g you? </p> Please DM us more info so we can advise furth
er. https://t.co/5pyLDJBC6r
```

AI VIET NAM
@aivietnam.edu.vn

1 – Text preprocessing

Problems?

➢ **Preprocessed Data**

```
We would like to help Sam, which number is
calling you? Please direct message us more
information so we can advise further.
```

➢ **Actual Data**

```
@82476 🤔 <p> We'd like to help Sam, which n
umber is caling you? </p> Please DM us more
 info so we can advise further. https://t.c
o/5pyLDJBC6r
```

Learned

Predict

Good or Not?

1 – Text preprocessing

Problems?

- ❑ Removal of URLs and HTML tags
- ❑ Text Standardizing
- ❑ Lowercasing
- ❑ Number and Punctuation Handling

- ❑ Removal Stop Words
- ❑ Removal Rare Words
- ❑ Handle Emoji and Emoticons
- ❑ Spelling Correction

- ❑ Tokenization
  - Sentence
  - Word
  - Character
  - Subwords
- ❑ Stemming
- ❑ Lemmatization

1 – Text preprocessing

1.1. Removal URLs, HTML Tags

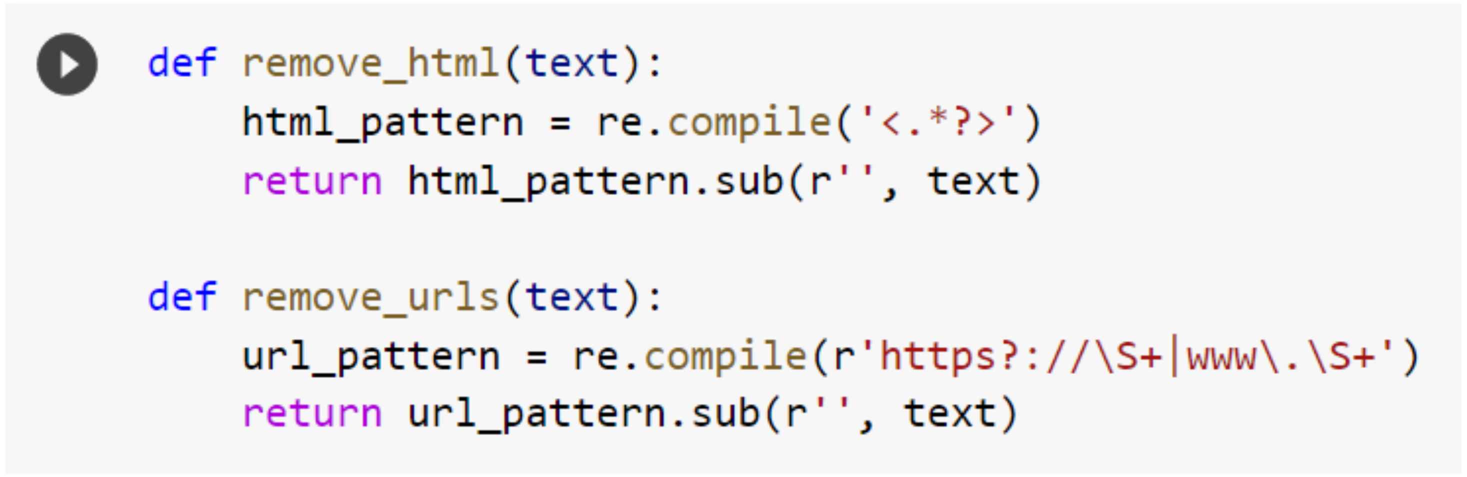
- Extract text based on the structure of an HTML document
- URLs: image links, reference links,…
- HTML tags: <p>..</p>, <div>…</div>,…

```
@AppleSupport causing the reply to be disregarded and the
tapped notification under the keyboard is opened😡😡😡
@82476 🤔 <p> We'd like to help Sam, which number is calin
g you? </p> Please DM us more info so we can advise furth
er. https://t.co/5pyLDJBC6r
```

1 – Text preprocessing

1.1. Removal URLs, HTML Tags

- Extract text based on the structure of an HTML document
- URLs: image links, reference links,…
- HTML tags: <p>..</p>, <div>…</div>,…

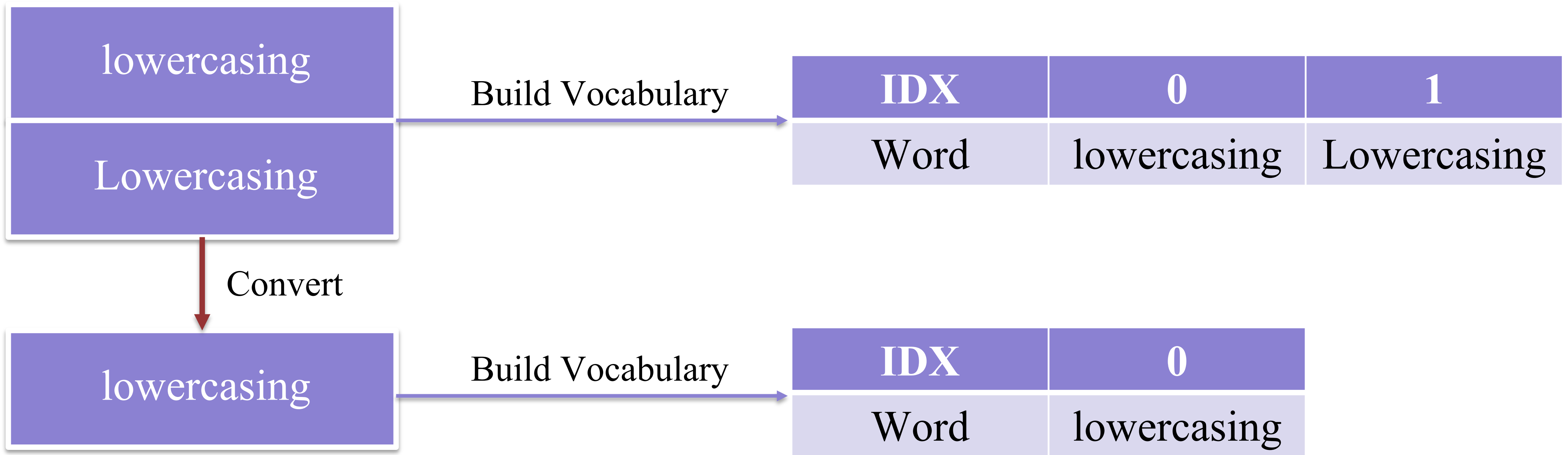


```
def remove_html(text):
    html_pattern = re.compile('<.*?>')
    return html_pattern.sub(r'', text)

def remove_urls(text):
    url_pattern = re.compile(r'https?://\S+|www\.\S+')
    return url_pattern.sub(r'', text)
```

1 – Text preprocessing

1.2. Lowercasing



IDX	0	1
Word	lowercasing	Lowercasing

IDX	0
Word	lowercasing

AI VIET NAM
@aivietnam.edu.vn

1 – Text preprocessing

1.2. Lowercasing

➢ Use lower() function in Python

```
@82476 🤔  We would like to help Sam, which number is caling you? Please direct message us more information so we can advise further.
```

```
@82476 🤔  we would like to help sam, which number is caling you? please direct message us more information so we can advise further.
```

AI VIET NAM
@aivietnam.edu.vn

1 – Text preprocessing

1.3. Standardizing

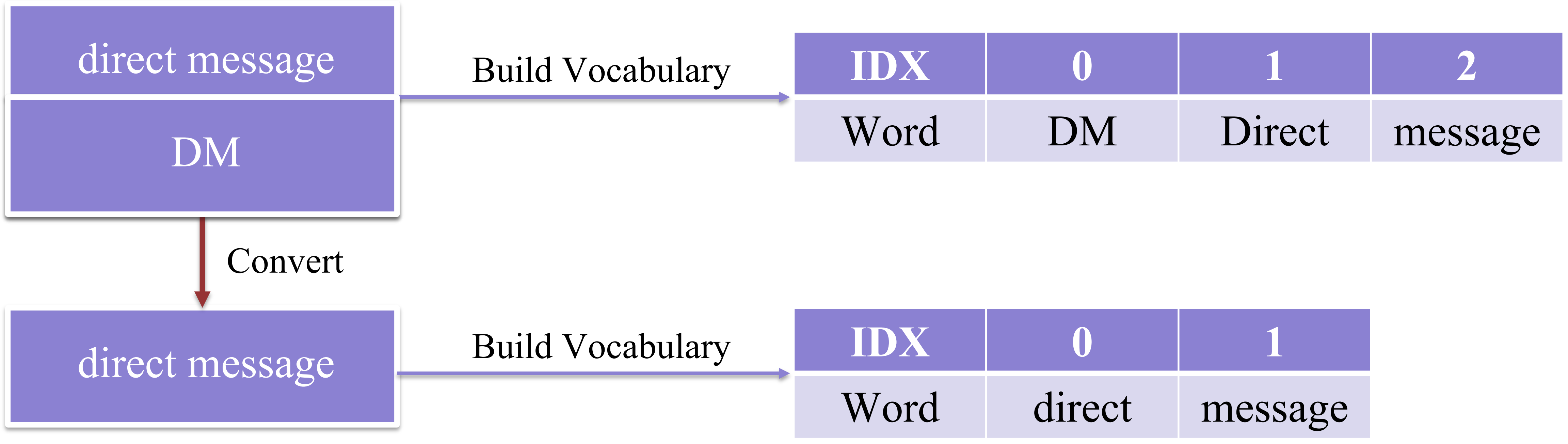
- Using short words and abbreviations to represent the same meaning

```
@AppleSupport causing the reply to be disregarded and the
tapped notification under the keyboard is opened😡😡😡
@82476 🤔 <p> We'd like to help Sam, which number is calin
g you? </p> Please DM us more info so we can advise furth
er. https://t.co/5pyLDJBC6r
```

1 – Text preprocessing

1.3. Standardizing

➢ Using short words and abbreviations to represent the same meaning



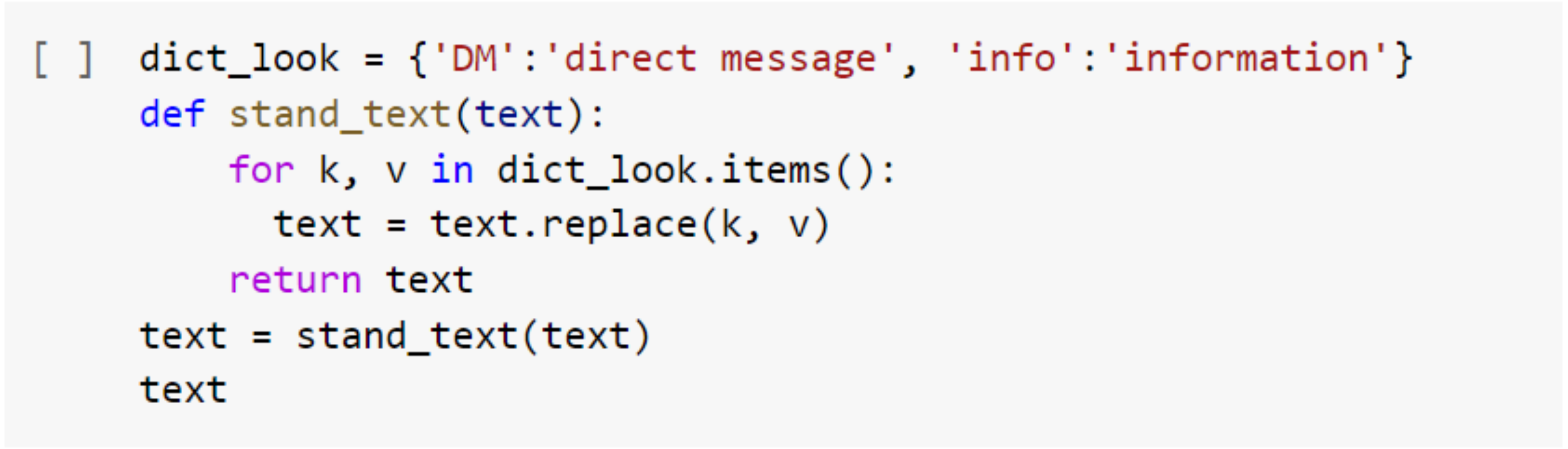
IDX	0	1	2
Word	DM	Direct	message

IDX	0	1
Word	direct	message

1 – Text preprocessing

1.3. Standardizing

- Using short words and abbreviations to represent the same meaning
- Build dictionary to look for short words and abbreviations



```
[ ]  dict_look = {'DM':'direct message', 'info':'information'}
     def stand_text(text):
         for k, v in dict_look.items():
           text = text.replace(k, v)
         return text
     text = stand_text(text)
     text
```

1 – Text preprocessing

1.3. Standardizing

- Using short words and abbreviations to represent the same meaning
- Build dictionary to look for short words and abbreviations

```
@82476 🤔  We'd like to help Sam,
which number is caling you?
Please DM us more info so we can
advise further.
```

```
@82476 🤔  We'd like to help Sam,
which number is caling you?
Please direct message us more
information so we can advise
further.
```

AI VIET NAM
@aivietnam.edu.vn

1 – Text preprocessing

1.3. Standardizing

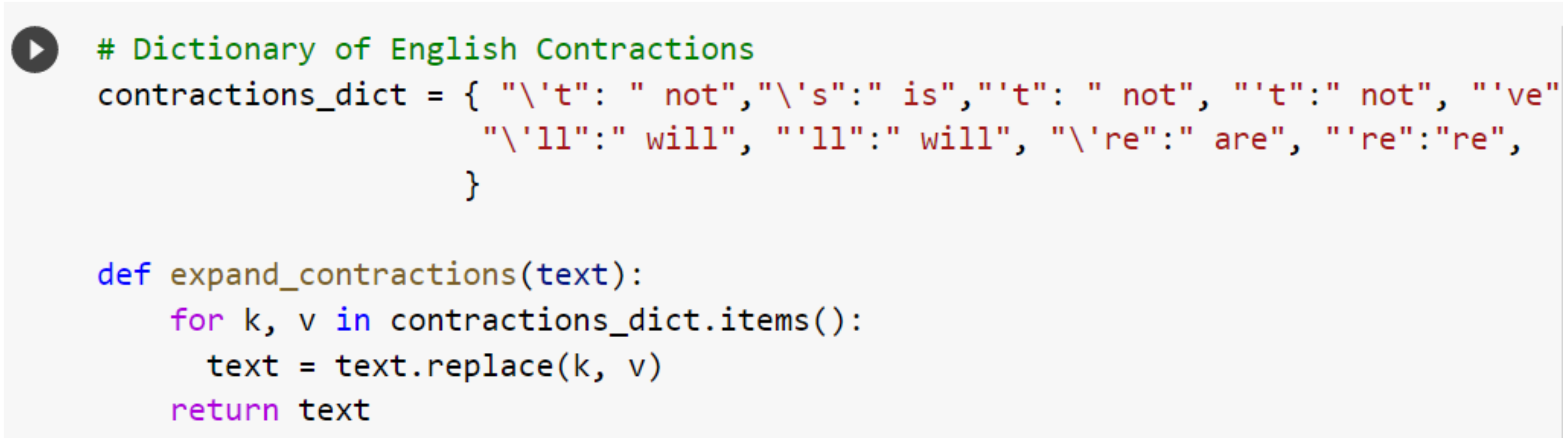
- Contractions: I’m, is’t, can’t,…

```
@AppleSupport causing the reply to be disregarded and the
tapped notification under the keyboard is opened😡😡😡
@82476 🤔 <p> We'd like to help Sam, which number is calin
g you? </p> Please DM us more info so we can advise furth
er. https://t.co/5pyLDJBC6r
```

1 – Text preprocessing

1.3. Standardizing

- Contractions: I’m, is’t, can’t,…
- Build dictionary or use library: contractions (pypi)



```
# Dictionary of English Contractions
contractions_dict = { "\'t": " not","\'s":" is","'t": " not", "'t":" not", "'ve"
                     "\'ll":" will", "'ll":" will", "\'re":" are", "'re":"re",
                    }

def expand_contractions(text):
    for k, v in contractions_dict.items():
      text = text.replace(k, v)
    return text
```

1 – Text preprocessing

1.3. Standardizing

- Contractions: I’m, is’t, can’t,…
- Build dictionary or use library: contractions (pypi)

@82476 🤔 We'd like to help Sam, which number is caling you? Please direct message us more information so we can advise further.

@82476 🤔 We would like to help Sam, which number is caling you? Please direct message us more information so we can advise further.

1 – Text preprocessing

1.4. Number and Punctuation Handling

@82476 🤔 We would like to help Sam, which number is caling you? Please direct message us more information so we can advise further.

	Removal	As token
Sam,	Sam	Sam ,
you?	you	You ?
further.	further	Further .

1 – Text preprocessing

1.4. Number and Punctuation Handling

@82476 🤔 We would like to help Sam, which number is caling you? Please direct message us more information so we can advise further.

Sam,
you?
further.

Removal

Text Classification

As token

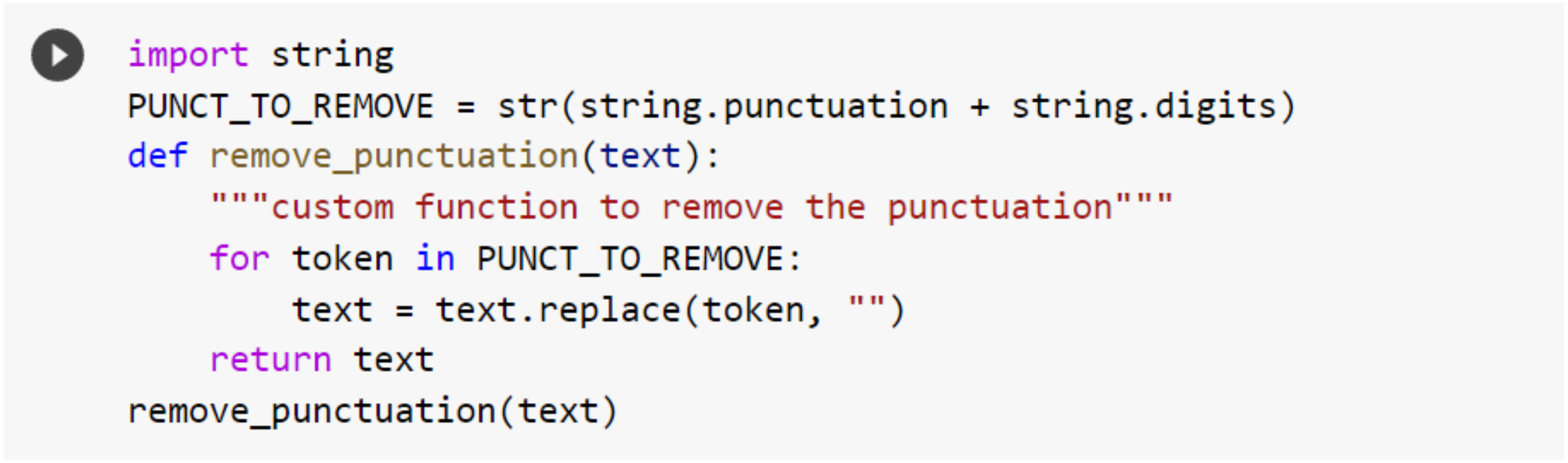
Machine Translation

POS, NER,…

1 – Text preprocessing

1.4. Number and Punctuation Handling

- Removal number and punctuation



```python
import string
PUNCT_TO_REMOVE = str(string.punctuation + string.digits)
def remove_punctuation(text):
    """custom function to remove the punctuation"""
    for token in PUNCT_TO_REMOVE:
        text = text.replace(token, "")
    return text
remove_punctuation(text)
```

1 – Text preprocessing

1.4. Number and Punctuation Handling

- Removal number and punctuation

```
@82476 🤔  We would like to help
Sam,  which number is caling you?
Please  direct  message  us  more
information  so  we  can  advise
further.
```

```
🤔  We would like to help Sam
which number is caling you  Please
direct message us more information
so we can advise further
```

1 – Text preprocessing

1.4. Number and Punctuation Handling

- Treat punctuation as token

```
PUNCT_TO_REMOVE = str(string.punctuation)

def convert_punc(text):
    """custom function to remove the punctuation"""
    for token in PUNCT_TO_REMOVE:
        text = text.replace(token, " " + token + " ")
    return text
```

AI VIET NAM
@aivietnam.edu.vn

1 – Text preprocessing

1.4. Number and Punctuation Handling

➢ Treat punctuation as token

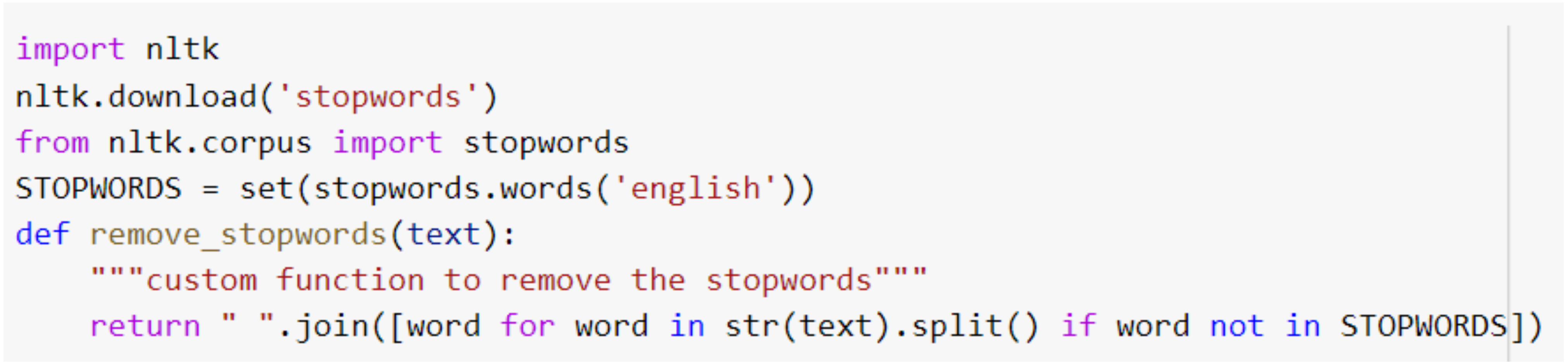
@82476 🤔 We would like to help Sam, which number is caling you? Please direct message us more information so we can advise further.

@ 82476 🤔 We would like to help Sam , which number is caling you ? Please direct message us more information so we can advise further .

1 – Text preprocessing

1.5. Removal Stop Words

- Stop words: common words that carry no meaning or less meaning compared to other keywords
- Focus on the important keywords
- English: a, an, the, that Vietnamese: à, ừ, vậy, thế,…



```
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
STOPWORDS = set(stopwords.words('english'))
def remove_stopwords(text):
    """custom function to remove the stopwords"""
    return " ".join([word for word in str(text).split() if word not in STOPWORDS])
```

1 – Text preprocessing

1.5. Removal Stop Words

- Stop words: common words that carry no meaning or less meaning compared to other keywords
- Focus on the important keywords
- English: a, an, the, that

Vietnamese: à, ừ, vậy, thế,…

```
@82476 🤔  We would like to help
Sam, which number is caling you?
Please direct message us more
information so we can advise
further.
```

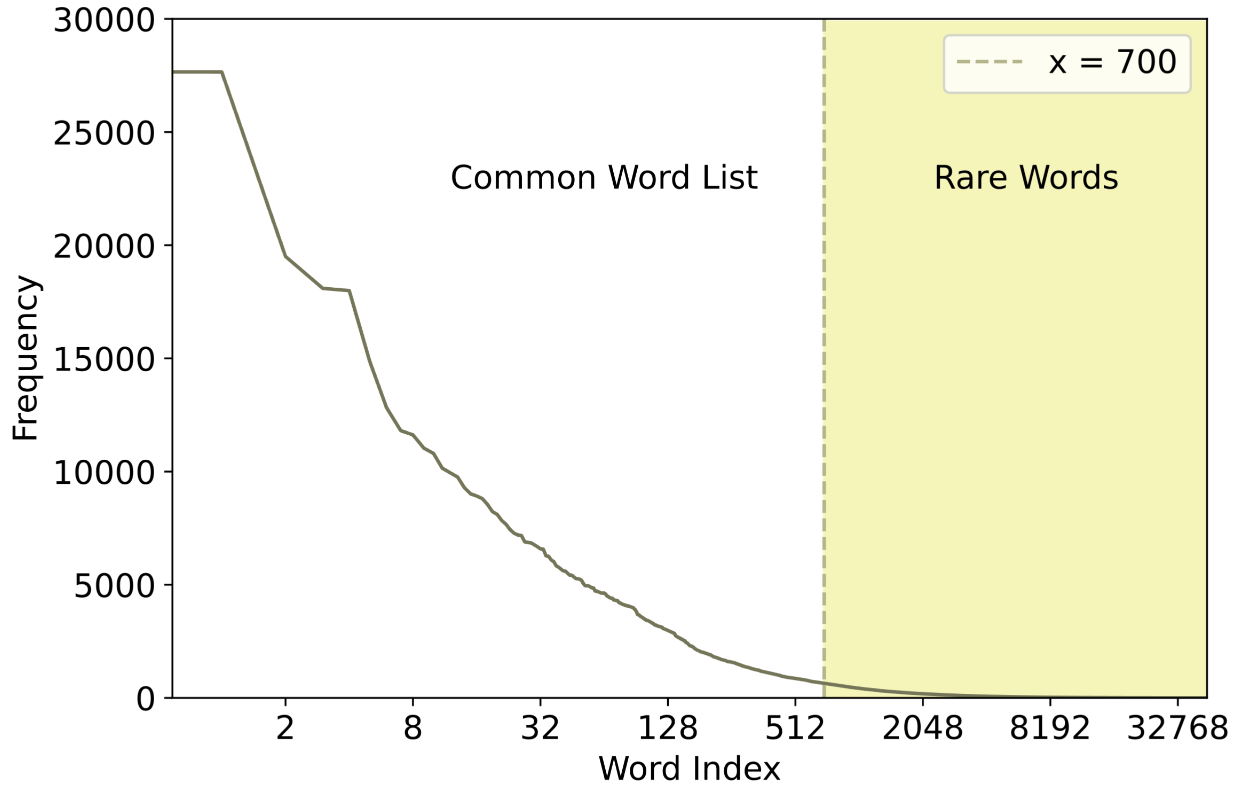
```
@82476 🤔 would like help sam,
number caling you? please direct
message us information advise
further.
```

AI VIET NAM
@aivietnam.edu.vn

1 – Text preprocessing

1.6. Removal Rare Words

- Rare words that appear only a few times in corpus
- Goal: focus on the important keywords
- Remove rare words based on their occurrence frequency



1 – Text preprocessing

1.7. Emoji and Emoticons Handling

- Emojis: ☮️ 🙂 ❤️…
- Emoticons: :-) :-(:-))) :-)

```
@AppleSupport causing the reply to be disregarded and the
tapped notification under the keyboard is opened 😡😡😡
@82476 🤔 <p> We'd like to help Sam, which number is calin
g you? </p> Please DM us more info so we can advise furth
er. https://t.co/5pyLDJBC6r :))
```

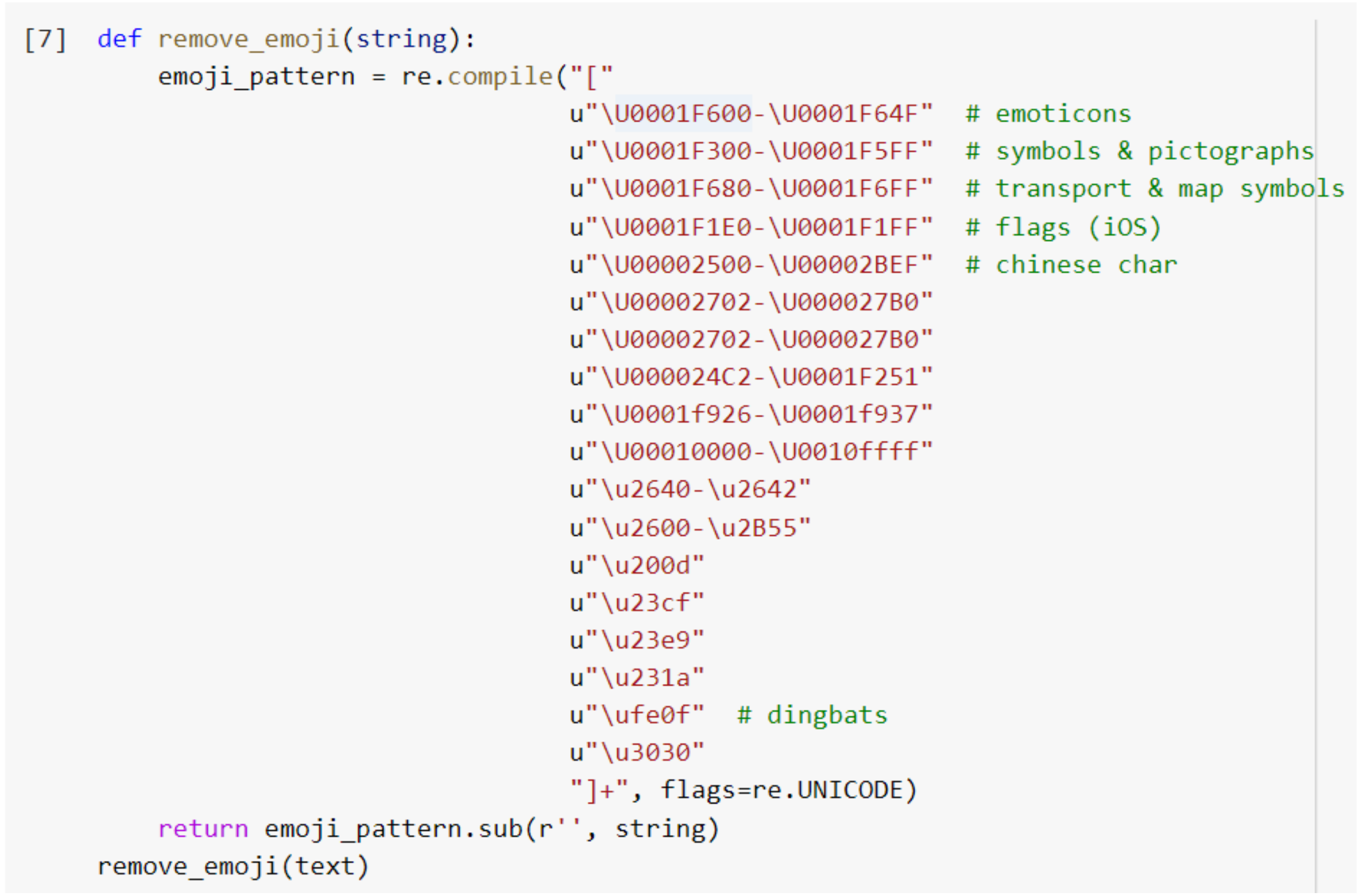
1 – Text preprocessing

1.7. Emoji and Emoticons Handling

- ➢ Most tasks:

removal emoji and emoticons

- ➢ Use RegEx (re)



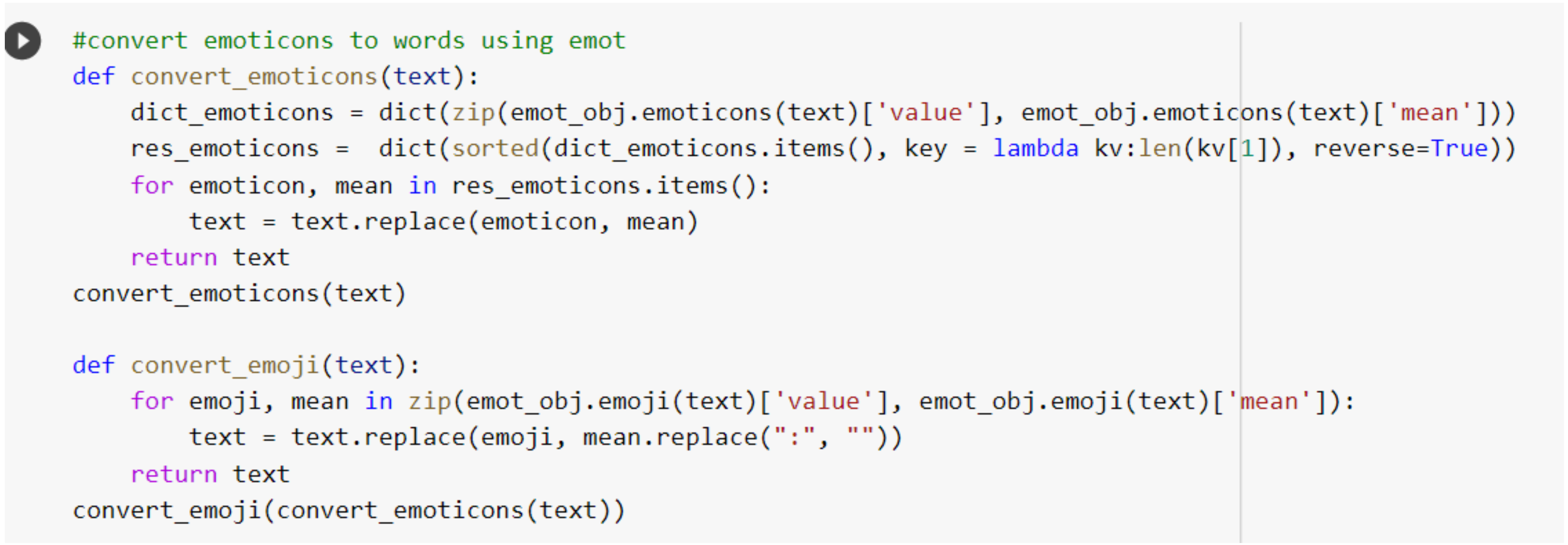
```
[7] def remove_emoji(string):
        emoji_pattern = re.compile("["
                                   u"\U0001F600-\U0001F64F"  # emoticons
                                   u"\U0001F300-\U0001F5FF"  # symbols & pictographs
                                   u"\U0001F680-\U0001F6FF"  # transport & map symbols
                                   u"\U0001F1E0-\U0001F1FF"  # flags (iOS)
                                   u"\U00002500-\U00002BEF"  # chinese char
                                   u"\U00002702-\U000027B0"
                                   u"\U00002702-\U000027B0"
                                   u"\U000024C2-\U0001F251"
                                   u"\U0001f926-\U0001f937"
                                   u"\U00010000-\U0010ffff"
                                   u"\u2640-\u2642"
                                   u"\u2600-\u2B55"
                                   u"\u200d"
                                   u"\u23cf"
                                   u"\u23e9"
                                   u"\u231a"
                                   u"\ufe0f"  # dingbats
                                   u"\u3030"
                                   "]+", flags=re.UNICODE)
        return emoji_pattern.sub(r'', string)
    remove_emoji(text)
```

AI VIET NAM
@aivietnam.edu.vn

1 – Text preprocessing

1.7. Emoji and Emoticons Handling

- Some tasks: convert emojis and emoticons to word.
- Example: :-) => happy, :-(=> sad,…



```
#convert emoticons to words using emot
def convert_emoticons(text):
    dict_emoticons = dict(zip(emot_obj.emoticons(text)['value'], emot_obj.emoticons(text)['mean']))
    res_emoticons =  dict(sorted(dict_emoticons.items(), key = lambda kv:len(kv[1]), reverse=True))
    for emoticon, mean in res_emoticons.items():
        text = text.replace(emoticon, mean)
    return text
convert_emoticons(text)

def convert_emoji(text):
    for emoji, mean in zip(emot_obj.emoji(text)['value'], emot_obj.emoji(text)['mean']):
        text = text.replace(emoji, mean.replace(":", ""))
    return text
convert_emoji(convert_emoticons(text))
```

1 – Text preprocessing

1.7. Emoji and Emoticons Handling

- Some tasks: convert emojis and emoticons to word.
- Example: :-) => happy, :-(=> sad,…

```
@82476 🤔 We would like to help Sam, which number is caling you? Please direct message us more information so we can advise further.
```

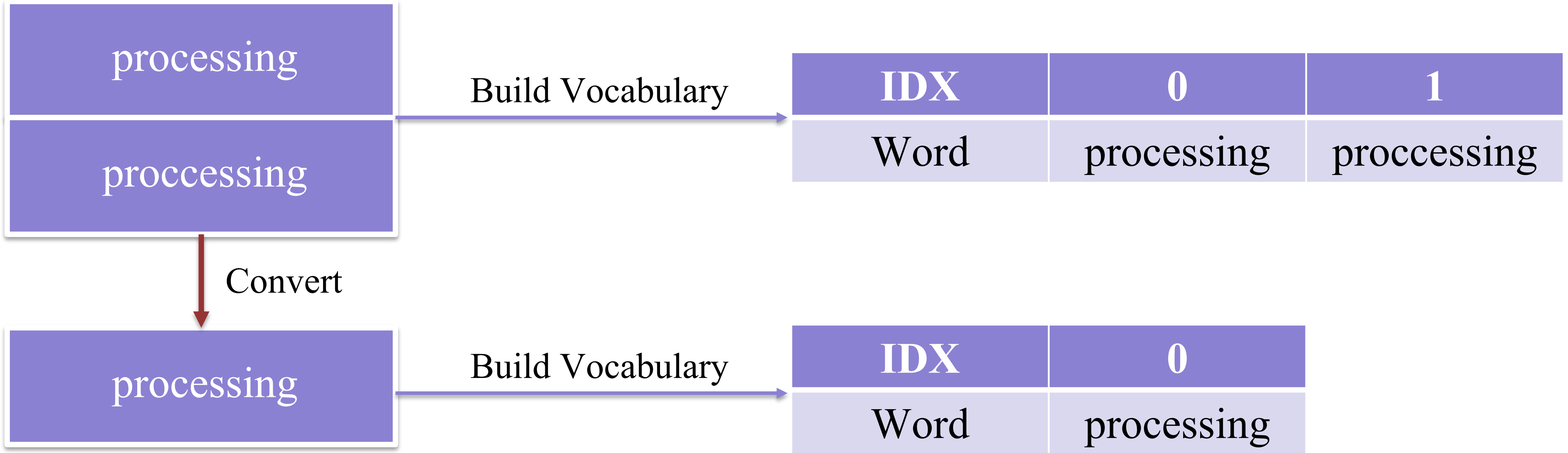
```
@82476 thinking face .We would like to help Sam, which number is caling you? Please direct message us more information so we can advise further.
```

AI VIET NAM
@aivietnam.edu.vn

1 – Text preprocessing

1.8. Spelling Correction

- Typo Errors
- Example: proccessing => Correct: processing



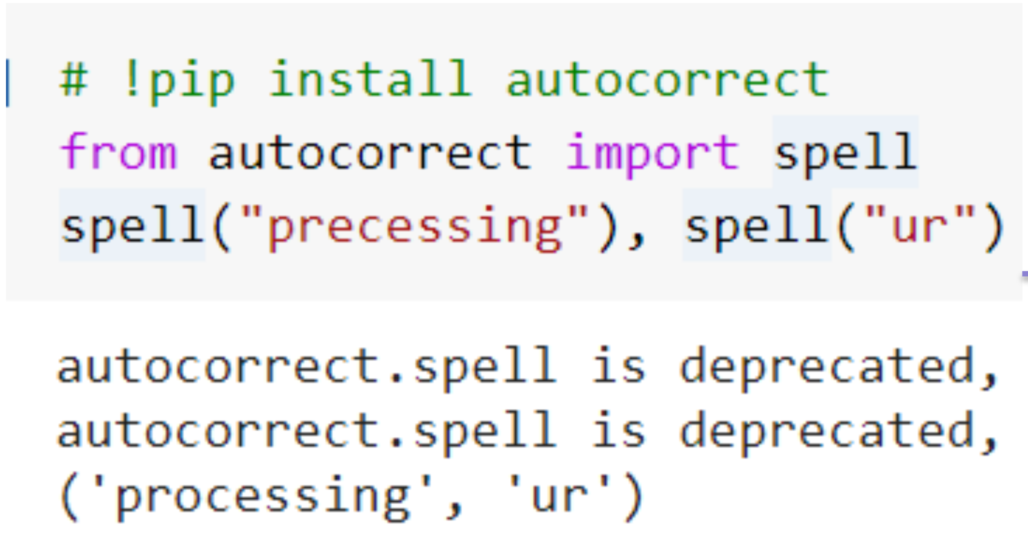
IDX	0	1
Word	processing	proccessing

IDX	0
Word	processing

1 – Text preprocessing

1.8. Spelling Correction

- Typo Errors
- Example: proccessing => Correct: processing



```
# !pip install autocorrect
from autocorrect import spell
spell("precessing"), spell("ur")
```

```
autocorrect.spell is deprecated,
autocorrect.spell is deprecated,
('processing', 'ur')
```

abbreviations should be handled before this step

1 – Text preprocessing

1.8. Spelling Correction

- Typo Errors
- Example: proccessing => Correct: processing

```
@82476 🤔  We would like to help
Sam, which number is caling you?
Please direct message us more
information so we can advise
further.
```

```
@82476 🤔  We would like to help
Sam, which number is calling you?
Please direct message us more
information so we can advise
further.
```

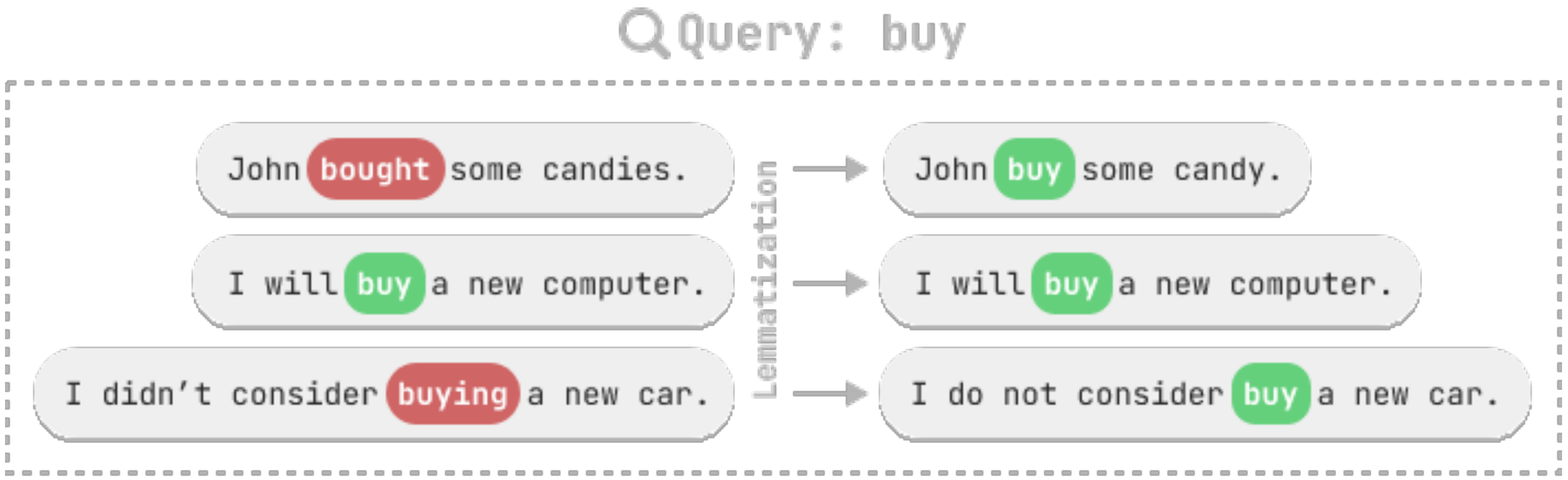
1 – Text preprocessing

1.9. Stemming and Lemmatization

- Lemmatization:

  words have the same root

  despite their surface differences


Query: buy
John bought some candies.
I will buy a new computer.
I didn't consider buying a new car.
Lemmatization
John buy some candy.
I will buy a new computer.
I do not consider buy a new car.

1 – Text preprocessing

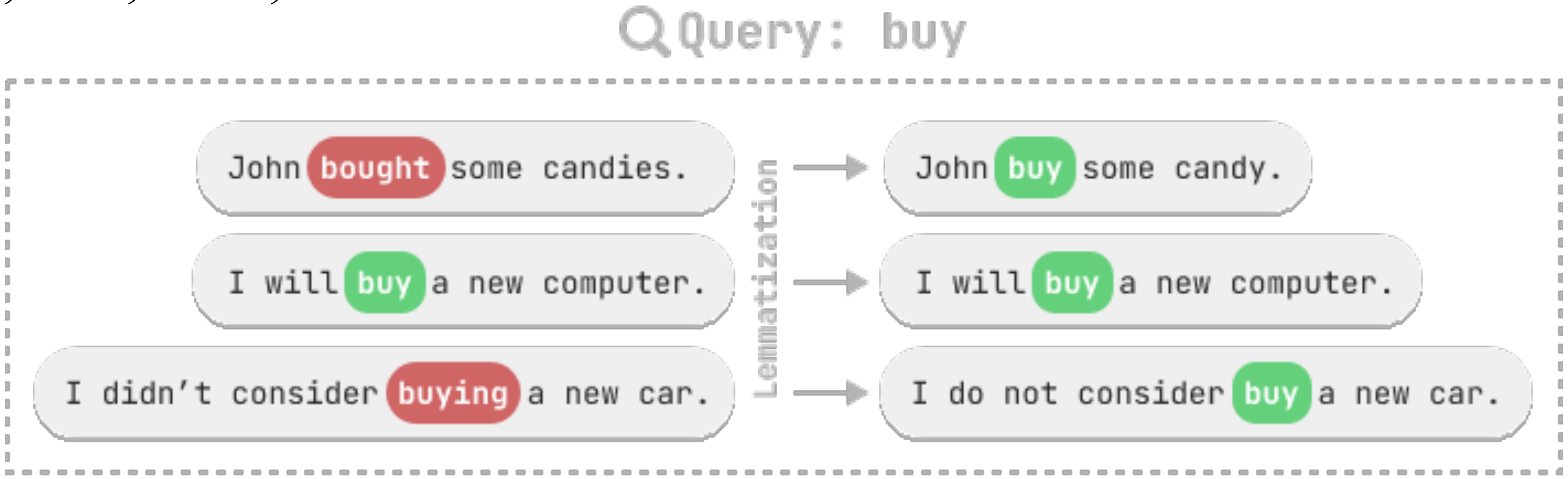
1.9. Stemming and Lemmatization

- Goal: convert => the same root

  am, is, are => be

  dinner, dinners => dinner

  car, cars, car's, cars' => car


Query: buy
John bought some candies.
I will buy a new computer.
I didn't consider buying a new car.
Lemmatization
John buy some candy.
I will buy a new computer.
I do not consider buy a new car.

1 – Text preprocessing

1.9. Stemming and Lemmatization

Morphological parsing

- Morphology: The small meaningful units that make up words
  - Stems: The core meaning-bearing units
  - Affixes: Parts that adhere to stems, often with grammatical functions
- Morphological Parsers:

$$\begin{tikzcd}
\text{Highest} \arrow[r] & \text{High} + \text{est} \\
\text{Higher} \arrow[r] & \text{High} + \text{er}
\end{tikzcd}$$

1 – Text preprocessing

1.9. Stemming and Lemmatization

Stemming

- Stemming – Simple Lemmatization

  Naïve version of morphological analysis

  Chopping off word-final stemming affixes

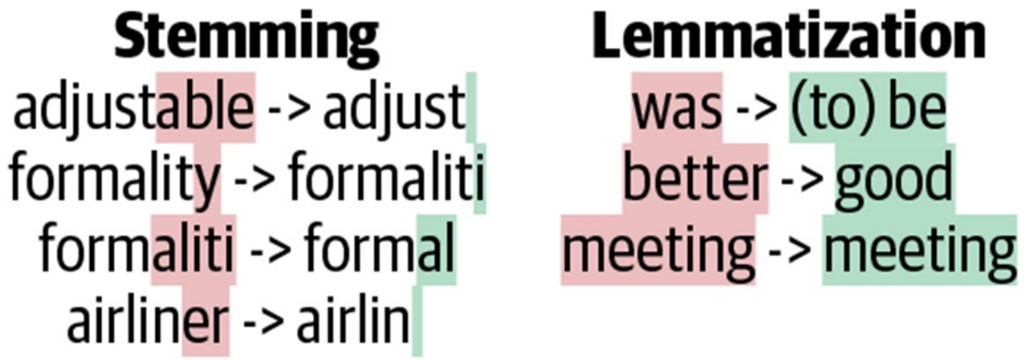
- The Porter Stemmer (1980): based on rewrite rules

…ational	→	…ate	relational => relate
…sses	→	…ss	grasses => grass

1 – Text preprocessing

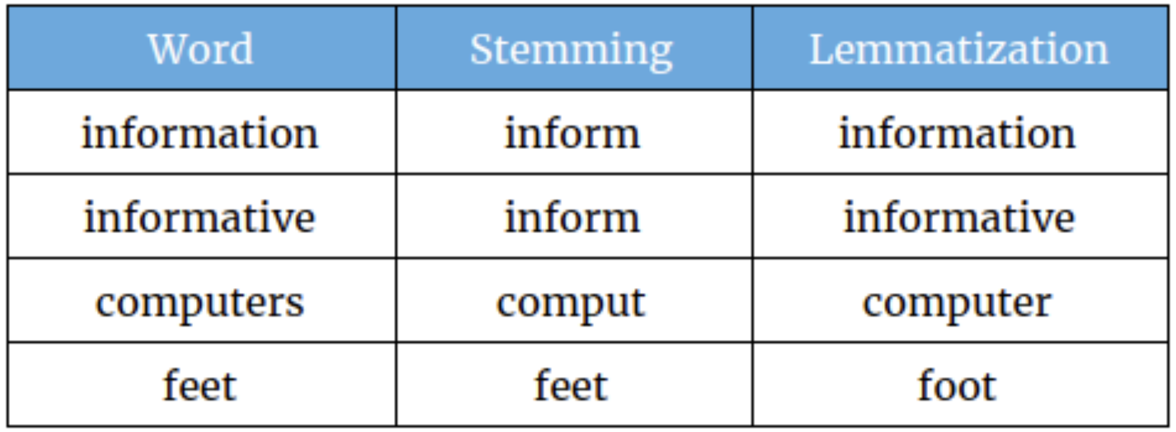
1.9. Stemming and Lemmatization

Compare stemming and lemmatization



Stemming
adjustable -> adjust
formality -> formaliti
formaliti -> formal
airliner -> airlin

Lemmatization
was -> (to) be
better -> good
meeting -> meeting



Word	Stemming	Lemmatization
information	inform	information
informative	inform	informative
computers	comput	computer
feet	feet	foot

2 – Tokenization

2.1. Sentence Tokenization

- Split paragraph, document into sentences
- Use RegEx or library: nltk, genism,… => nltk.sent_tokenize()

Input Text

Tokenization is one of the first step in any NLP pipeline. Tokenization is nothing but splitting the raw text into small chunks of words or sentences, called tokens

Sentence Tokenization

Tokenization is one of the first step in any NLP pipeline.

Tokenization is nothing but splitting the raw text into small chunks of words or sentences, called tokens

2 - Tokenization

2.2. Word Tokenization

Word level

The	most	eager	is	Oregon	which	is	enlisting	5,000	drivers	in	the	country

Char level

T	h	e	m	o	s	t	e	a	g	e	r	i	s	O	r	e	g	…

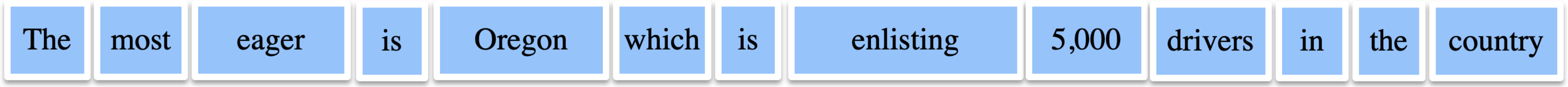
Sub-word level

The	most	e	ager	is	O	reg	on	which	is	en	listing	5,000	drivers	in	the	country

2 - Tokenization

2.2. Word Tokenization

Word level



❑ Tokenize based on a delimiter as space

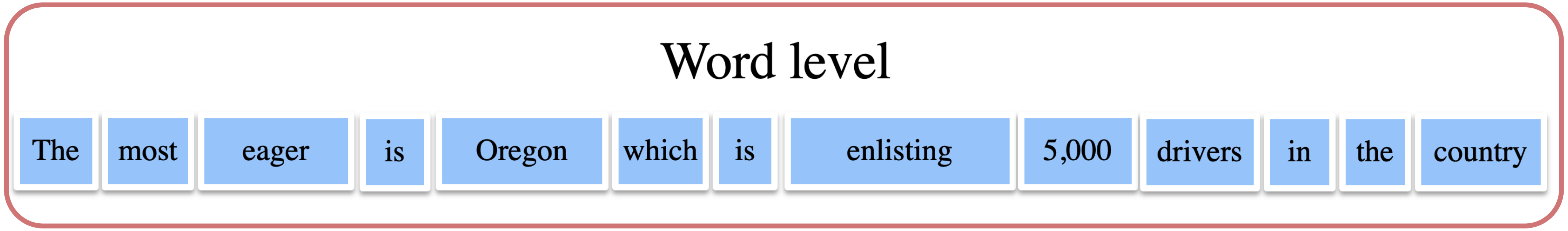
❑ Using custom RegEx or split()

```
text = "i do not like coffee. and you?"
text.split()
```

```
['i', 'do', 'not', 'like', 'coffee.', 'and', 'you?']
```

2 - Tokenization

2.2. Word Tokenization



❖ **Problem:** punctuations occurs word internally

prices ($12.34), names (Mr.Bean), percentage (100%), dates (01/09/2021).

```python
import re, string
text = "Mr.Bean does not like coffee, $12.34, 100%."
print(re.findall(r"\w+|[.,!?;]", text))
```

```
['Mr', '.', 'Bean', 'does', 'not', 'like', 'coffee', ',', '12', '.', '34', ',', '100', '.']
```

AI VIET NAM
@aivietnam.edu.vn

2 - Tokenization

2.2. Word Tokenization

Word level

The	most	eager	is	Oregon	which	is	enlisting	5,000	drivers	in	the	country

❖ **Problem:** punctuations occurs word internally

prices ($12.34), names (Mr.Bean), percentage (100%), dates (01/09/2021).

Penn Tree Tokenization

```
from nltk.tokenize import TreebankWordTokenizer
text = "Mr.Bean does not like coffee, $12.34, 100%."
print(TreebankWordTokenizer().tokenize(text))
```

```
['Mr.Bean', 'does', 'not', 'like', 'coffee', ',', '$', '12.34', ',', '100', '%', '.']
```

2 - Tokenization

2.2. Word Tokenization

Sub-word level

The	most	e	ager	is	O	reg	on	which	is	en	listing	5,000	drivers	in	the	country

❖ **Problem:** based on: morphological parsing

corpus		vocabulary
5	l o w </w>	'l', 'o', 'w', '</w>', 'e', 's', 't', 'i', 'd', 'er</w>', 'new'
2	l o w e s t </w>	len: 11
6	ne w er</w>	
3	w i d er</w>	
2	ne w </w>	

2 - Tokenization

2.2. Word Tokenization

Sub-word level

The	most	e	ager	is	O	reg	on	which	is	en	listing	5,000	drivers	in	the	country

❖ Three common algorithms:

- ➢ Byte-Pair Encoding (BPE) (Sennrich et al., 2016) used by GPT-2 and RoBERTa,...
- ➢ Unigram language modeling tokenization (Kudo, 2018) used by XLNET, ALBERT,...
- ➢ WordPiece (Schuster and Nakajima, 2012) used by BERT, DistilBERT,...

Thanks!

Any questions?