

Tokenization in Natural Language Processing

2/2565: FRA501 Introduction to Natural Language Processing with Deep learning
Week 02

Paisit Khanarsa, Ph.D.

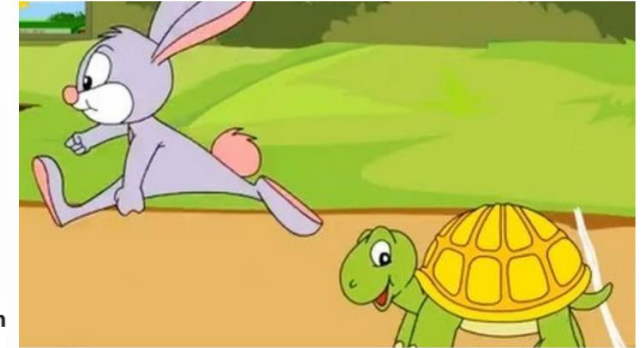
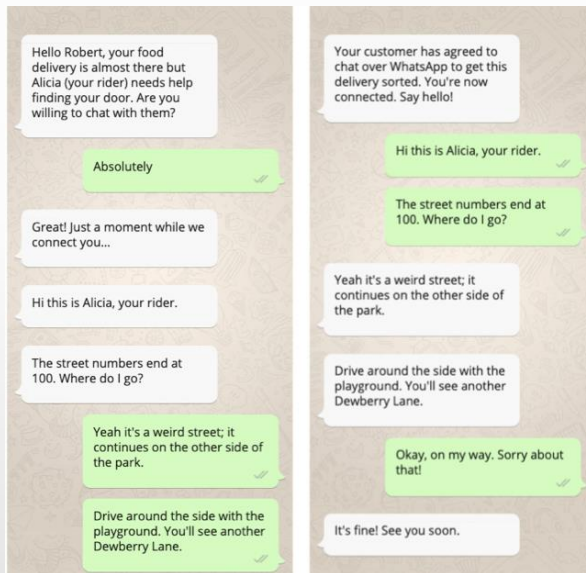
Institute of Field Robotics (FIBO), King Mongkut's University of Technology Thonburi

Outlines

- Why we do tokenization?
- Tokenization approaches
 - Dictionary-based
 - Longest matching
 - Maximal matching
 - Machine learning and deep learning-based
- Demo

Why we do tokenization?

- Text as a stream of characters



“Once upon a time a Tortoise and a Rabbit had an
faster.

They decided to settle the argument with a race. 1 **Rabbit and tortoise. Story**
and started off the race. The rabbit shot ahead and **continues.**

time. Then seeing that he was far ahead of the tortoise, he thought he'd sit
under a tree for some time and relax before continuing the race. He sat
under the tree and soon fell asleep. The tortoise plodding on overtook him
and

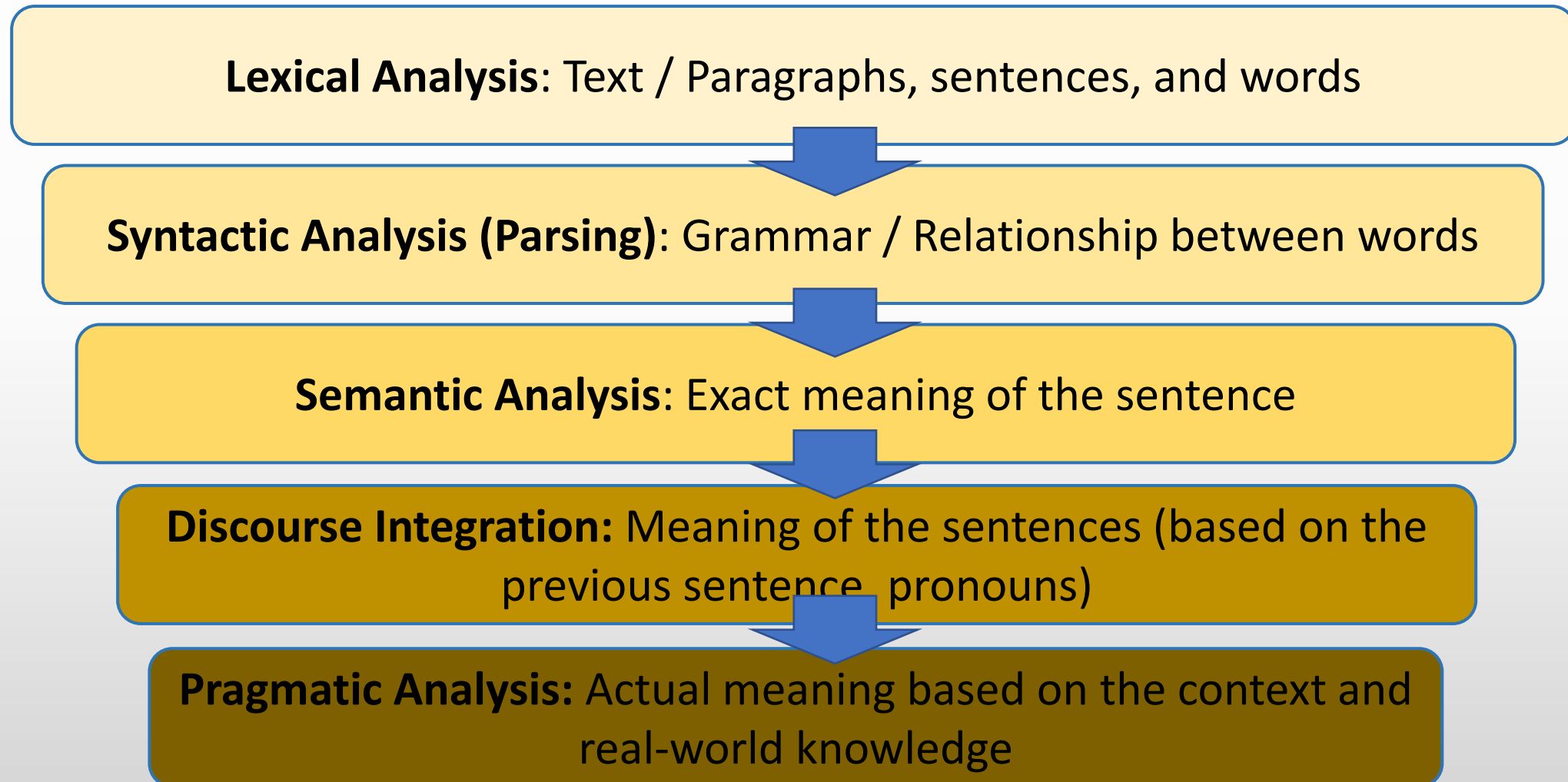
soon finished the race, emerging as the undisputed champ. The rabbit woke
up and realized that he'd lost the race.”

The moral of the story is that slow and steady wins the race. This is the version of
the story that we've all grown up with.

But then recently, the story still continues:

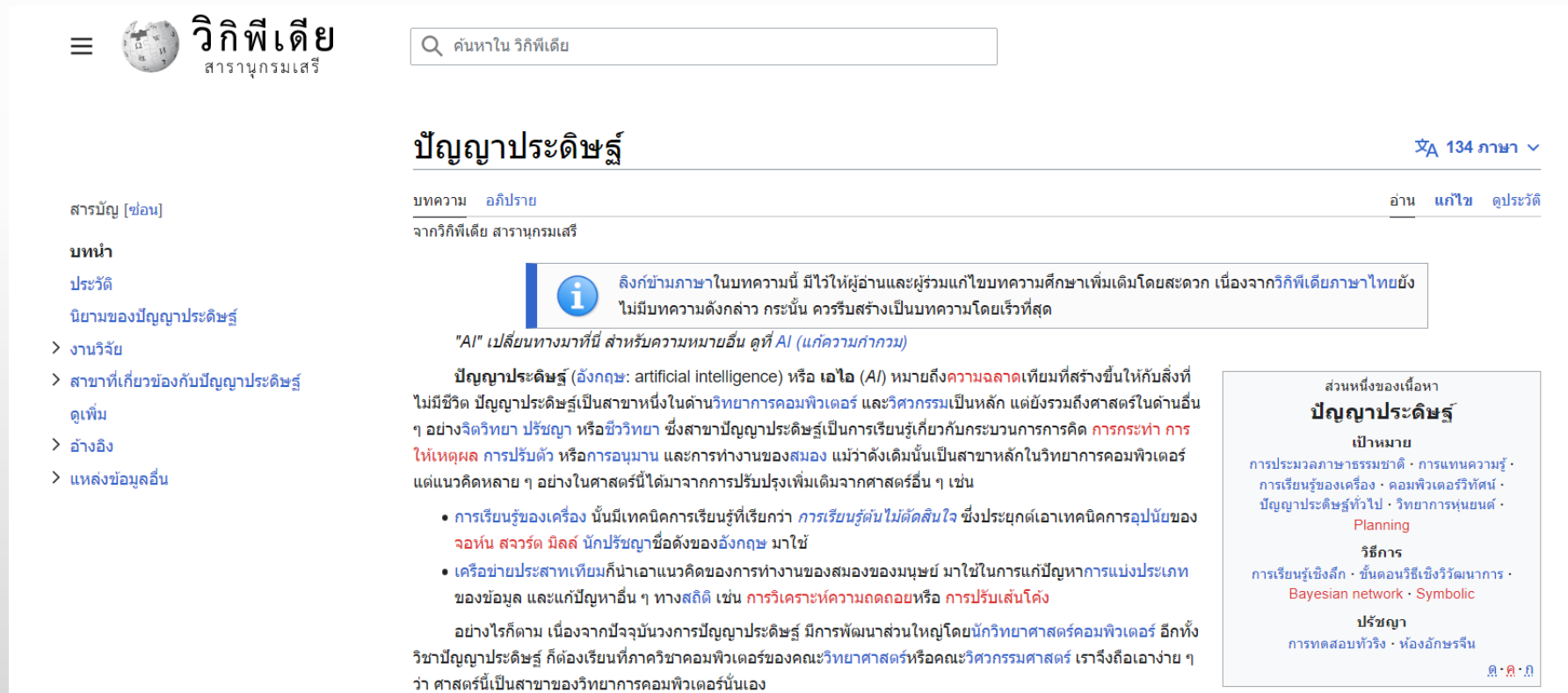
- How to understand the meaning of texts
 - Break into words (assign meaning to words) ---> **Word tokenization**
 - Break into sentences (put word meaning back to sentence meaning)

Level of understand in NLP



Tokenization – Thai (cont.)

- Thai has no space between words
- Thai has no clear sentence boundaries



วิกิพีเดีย สารานุกรมเสรี

ค้นหาใน วิกิพีเดีย

ปัญญาประดิษฐ์

บทความ อภิปราย

จากวิกิพีเดีย สารานุกรมเสรี

อ่าน แก้ไข ดูประวัติ

ลิงก์ข้ามภาษาในบทความนี้ มีไว้ให้ผู้อ่านและผู้ร่วมแก้ไขบทความศึกษาเพิ่มเติมโดยสะดวก เนื่องจากวิกิพีเดียภาษาไทยยังไม่มีบทความดังกล่าว กระนั้น ควรรีบสร้างเป็นบทความโดยเร็วที่สุด

"AI" เปลี่ยนทางมาที่นี่ สำหรับความหมายอื่น ดูที่ AI (แก้ความกำกวม)

ปัญญาประดิษฐ์ (อังกฤษ: artificial intelligence) หรือ **เอไอ** (AI) หมายถึง**ความฉลาด**เทียมที่สร้างขึ้นให้กับสิ่งที่ไม่มีชีวิต ปัญญาประดิษฐ์เป็นสาขาหนึ่งในด้านวิทยาการคอมพิวเตอร์ และวิศวกรรมเป็นหลัก แต่ยังรวมถึงศาสตร์ในด้านอื่น ๆ อย่างจิตวิทยา ปรัชญา หรือชีววิทยา ซึ่งสาขาปัญญาประดิษฐ์เป็นการเรียนรู้เกี่ยวกับกระบวนการการคิด การกระทำ การให้เหตุผล การปรับตัว หรือการอนุมาน และการทำงานของสมอง แม้ว่าดังเดิมนั้นเป็นสาขาหลักในวิทยาการคอมพิวเตอร์ แต่แนวคิดหลาย ๆ อย่างในศาสตร์นี้ได้มาจากการปรับปรุงเพิ่มเติมจากศาสตร์อื่น ๆ เช่น

- การเรียนรู้ของเครื่อง นั้นมีเทคนิคการเรียนรู้ที่เรียกว่า *การเรียนรู้ต้นไม้ตัดสินใจ* ซึ่งประยุกต์เอาเทคนิคการอุปนัยของจอห์น สจวร์ต มิลล์ นักปรัชญาชื่อดังของอังกฤษ มาใช้
- เครือข่ายประสาทเทียมก็นำเอาแนวคิดของการทำงานของสมองของมนุษย์ มาใช้ในการแก้ปัญหาการแบ่งประเภทของข้อมูล และแก้ปัญหาอื่น ๆ ทางสถิติ เช่น การวิเคราะห์ความถดถอยหรือ การปรับเส้นโค้ง

อย่างไรก็ตาม เนื่องจากปัจจุบันวงการปัญญาประดิษฐ์ มีการพัฒนาส่วนใหญ่โดยนักวิทยาศาสตร์คอมพิวเตอร์ อีกทั้งทั้งวิชาปัญญาประดิษฐ์ ก็ต้องเรียนที่ภาควิชาคอมพิวเตอร์ของคณะวิทยาศาสตร์หรือคณะวิศวกรรมศาสตร์ เราจึงถือเอาง่าย ๆ ว่า ศาสตร์นี้เป็นสาขาของวิทยาการคอมพิวเตอร์นั่นเอง

ส่วนหนึ่งของเนื้อหา

ปัญญาประดิษฐ์

เป้าหมาย

การประมวลภาษาธรรมชาติ · การแทนความรู้ · การเรียนรู้ของเครื่อง · คอมพิวเตอร์วิทัศน์ · ปัญญาประดิษฐ์ทั่วไป · วิทยาการหุ่นยนต์ · Planning

วิธีการ

การเรียนรู้เชิงลึก · ขั้นตอนวิธีเชิงวิวัฒนาการ · Bayesian network · Symbolic

ปรัชญา

การทดสอบทัวริง · ห้องอภิศรี

ด · ค · ก

<https://th.wikipedia.org/wiki/%E0%B8%9B%E0%B8%B1%E0%B8%8D%E0%B8%8D%E0%B8%B2%E0%B8%9B%E0%B8%A3%E0%B8%B0%E0%B8%94%E0%B8%B4%E0%B8%A9%E0%B8%90%E0%B9%8C>

Tokenization – Thai (cont.)

- Social media texts



Tokenization – Thai (cont.)

- Many word boundaries depend on the context (meaning)

ตา กลม **VS** ตาก ลม

- Even amongst Thais the definition of word boundary is unclear
 - Needs a consensus when designing a corpus
 - Sometimes depends on the application

Tokenization - English

- Space is usually not enough
 - aren't
 - are + n't
 - aren't
 - arent
 - aren + t
 - are + not
 - San Francisco
 - San + Francisco
 - San Francisco
- Hyphens
 - The New York-based co-operation was fine-tuning forty-two K-9-like models
- Social media texts
 - @SentimentSymp: can't wait for the Nov 9 #Sentiment talks! YAAAAAY!!! >:-D <http://sentimentsymposium.com/>.

Tokenization – English (cont.)

- Social media texts

- @SentimentSymp: can't wait for the Nov 9 #Sentiment talks! YAAAAAY!!! >:-D <http://sentimentsymposium.com/>.

Hand-crafted for tweets

```
@sentimentsymp
:
can't
wait
for
the
Nov_09
#sentiment
talks
!
YAAAY
!
!
!
>:-D
http://sentimentsymposium.com/
.
```

```
@
SentimentSymp
:
ca
n't
wait
for
the
Nov
9
#
Sentiment
talks
!
YAAAAAY
!
!
!
>;
;
:
-D
http
:
//sentimentsymposium.com/
.
```

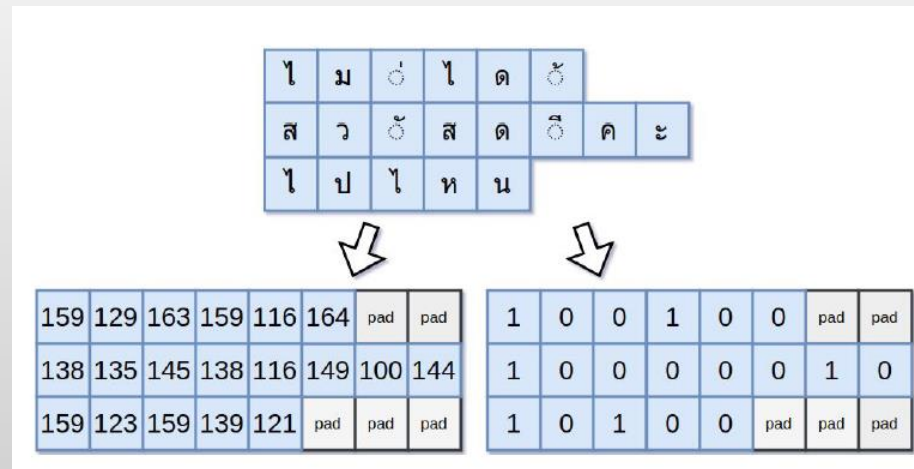
131

Standard tokenizer
(Stanford tokenizer)

Tokenization approaches

- Dictionary-based ---> Perform by scanning a string and match each sub string against words from **dictionary**.
 - Longest matching
 - Maximal matching
- Machine learning and deep learning-based ---> Change word to latent vector by training word features

http://web.archive.org/web/20181005101442/https://sertiscorp.com/thai-word-segmentation-with-bi-directional_rnn/



Longest Matching

- Scan a sentence from left to right
- Keep finding a word from the starting point, until no word matched (longest), then move to the next point
- Backtrack if current segmentation leads to an un-segmentable chunk

Example	Steps
ป้ายกลับรถ	Start scanning with “ป” as the starting point
ป้ายกลับรถ	Keep scanning
ป้าย / กลับรถ	No more words start with “ป้าย”, move to the next point
....
ป้าย / กลับ / รถ	Finish

Maximal Matching

- Generate all possible segmentations
- Select the segmentations with **the fewest words**

ไปห้ามเหสี

Longest Matching -----> ไป | ห้าม | เห | สี 4 words

Maximal Matching -----> ไป | หา | มเหสี 3 words

Maximal Matching

- Generate all possible segmentations
- Select the segmentations with **the fewest words**

- Maximal matching can be done using **dynamic programming**.
- Let $d(i,j)$ be the function which returns number of the fewest word possible with the **last word starts with i^{th} character (row) and ends with j^{th} character (column)**. It can be defined as:

$$d(i, j) = \begin{cases} 1 & \text{if } i=1 \text{ and } c[1, j] \text{ is in the dictionary.} \\ 1 + \min_{k=1 \dots i-1} d(k, i-1) & \text{if } c[i..j] \text{ is in the dictionary.} \\ \infty, & \text{otherwise.} \end{cases}$$

when $c[i..j]$ is a string of word in the sentence (assume it is started at index 1) and the base case is $d(1,1) = 1$.

Maximal Matching example

Input: ไปห้ามเหสี

Dict: ไป, ป, ห, า, ม, ะ, ส, ี, ไป, หา, ห้าม, เห, สี, มเหสี

Each $d[i, j]$ stores number of the fewest words, with the latest word starting at i and ending at j

	ไป	ป	ห	า	ม	ะ	ห	ส	ี
	1	2	3	4	5	6	7	8	9
1	1	1	∞	∞	∞	∞	∞	∞	∞
2		2	∞	∞	∞	∞	∞	∞	∞
3			2	2	2	∞	∞	∞	∞
4				3	∞	∞	∞	∞	∞
5					3	∞	∞	∞	3
6						3	3	∞	∞
7							4	∞	∞
8								4	4
9									5

Maximal Matching example

Input: ไปห้ามเหสี

Dict: ไป, ป, ห, า, ม, เ, ส, ี, ไป, หา, ห้าม, เห, สี, มเหสี

Each $d[i, j]$ stores number of the fewest words, with the latest word starting at i and ending at j

	ไป	ป	ห	า	ม	เ	ห	ส	ี
	1	2	3	4	5	6	7	8	9
1	1	1	∞	∞	∞	∞	∞	∞	∞
2		2	∞	∞	∞	∞	∞	∞	∞
3			2	2	2	∞	∞	∞	∞
4				3	∞	∞	∞	∞	∞
5					3	∞	∞	∞	3
6						3	3	∞	∞
7							4	∞	∞
8								4	4
9									5

Maximal Matching example

Input: ไปห้ามเหสี

Dict: ไป, ป, ห, า, ม, เ, ส, ี, ไป, หา, ห้าม, เห, สี, มเหสี

Each $d[i, j]$ stores number of the fewest words, with the latest word starting at i and ending at j

	ไป	ป	ห	า	ม	เ	ห	ส	ี
	1	2	3	4	5	6	7	8	9
1	1	1	∞	∞	∞	∞	∞	∞	∞
2		2	∞	∞	∞	∞	∞	∞	∞
3			2	2	2	∞	∞	∞	∞
4				3	∞	∞	∞	∞	∞
5					3	∞	∞	∞	3
6						3	3	∞	∞
7							4	∞	∞
8								4	4
9									5

Maximal Matching example

Input: ไปห้ามเหสี

Dict: ไป, ป, ห, า, ม, เ, ส, ี, ไป, หา, ห้าม, เห, สี, มเหสี

Each $d[i, j]$ stores number of the fewest words, with the latest word starting at i and ending at j

	ไป	ป	ห	า	ม	เ	ห	ส	ี
	1	2	3	4	5	6	7	8	9
1	1	1	∞	∞	∞	∞	∞	∞	∞
2		2	∞	∞	∞	∞	∞	∞	∞
3			2	2	2	∞	∞	∞	∞
4				3	∞	∞	∞	∞	∞
5					3	∞	∞	∞	3
6						3	3	∞	∞
7							4	∞	∞
8								4	4
9									5

Maximal Matching example

Input: ไปห้ามเหสี

Dict: ไป, ป, ห, า, ม, เ, ส, ี, ไป, หา, ห้าม, เห, สี, มเหสี

Each $d[i, j]$ stores number of the fewest words, with the latest word starting at i and ending at j

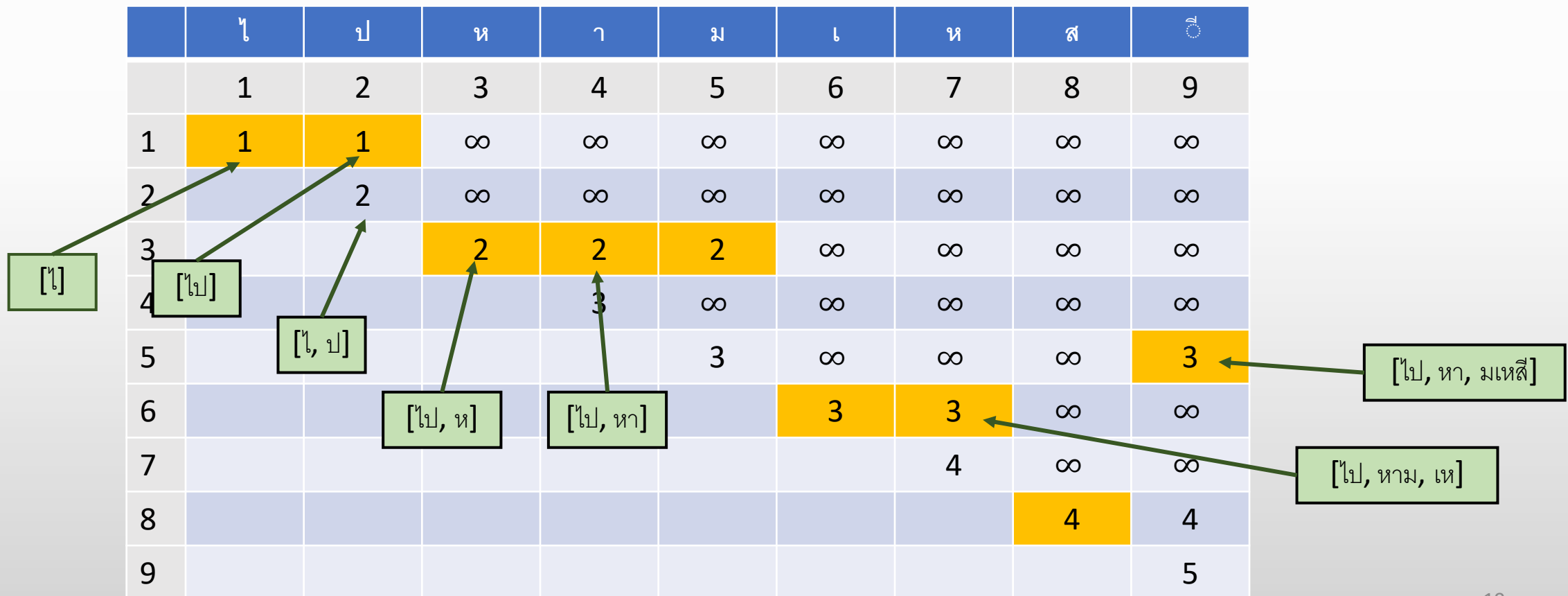
	ไป	ป	ห	า	ม	เ	ห	ส	ี
	1	2	3	4	5	6	7	8	9
1	1	1	∞	∞	∞	∞	∞	∞	∞
2		2	∞	∞	∞	∞	∞	∞	∞
3			2	2	2	∞	∞	∞	∞
4				3	∞	∞	∞	∞	∞
5					3	∞	∞	∞	3
6						3	3	∞	∞
7							4	∞	∞
8								4	4
9									5

Maximal Matching example

Input: ไปห้ามเหสี

Dict: ไป, ป, ห, า, ม, เ, ส, ี, ไป, หา, ห้าม, เห, สี, มเหสี

Each $d[i, j]$ stores number of the fewest words, with the latest word starting at i and ending at j



Maximal Matching example

Input: ไปห้ามเหสี

Dict: ไ, ป, ห, า, ม, ะ, ส, ี, ไป, หา, หาม, เห, ลี, มเหสี

Each $d[i, j]$ stores number of the fewest words, with the latest word starting at i and ending at j

The grid shows the following values (row, column):

	ไ	ป	ห	า	ม	เ	ห	ส	ี
1	1	1	∞	∞	∞	∞	∞	∞	∞
2		2	∞	∞	∞	∞	∞	∞	∞
3			2	2	2	∞	∞	∞	∞
4				3	∞	∞	∞	∞	∞
5					3	∞	∞	∞	3
6						3	3	∞	∞
7							4	∞	∞
8								4	4
9									5

Backtracking path (green boxes and arrows):

- Start at (5, 10) with path ["ไป, หา, มเหสี"]
- Move to (6, 7) with path ["ไป, หาม, เห"]
- Move to (3, 4) with path ["ไป, ห"]
- Move to (1, 2) with path ["ไป"]

Machine learning and deep learning-based

- Can represent the meaning of characters, words, sentences, etc. by training data.
- Thai-Tokenization

TABLE II
RESULTS OF THE SIX BEST TEAMS

Type of participants	F-Measure (%)	Time (mm:ss)
<i>Non-Students^a</i>	97.94937	00:47
<i>Non-Students</i>	97.84097	02:46
<i>Non-Students</i>	97.18822	00:26
<i>Bachelor Students^b</i>	95.78162	01:08
<i>Master Students</i>	95.56670	12:14
<i>PhD+Master Students</i>	92.02067	02:28

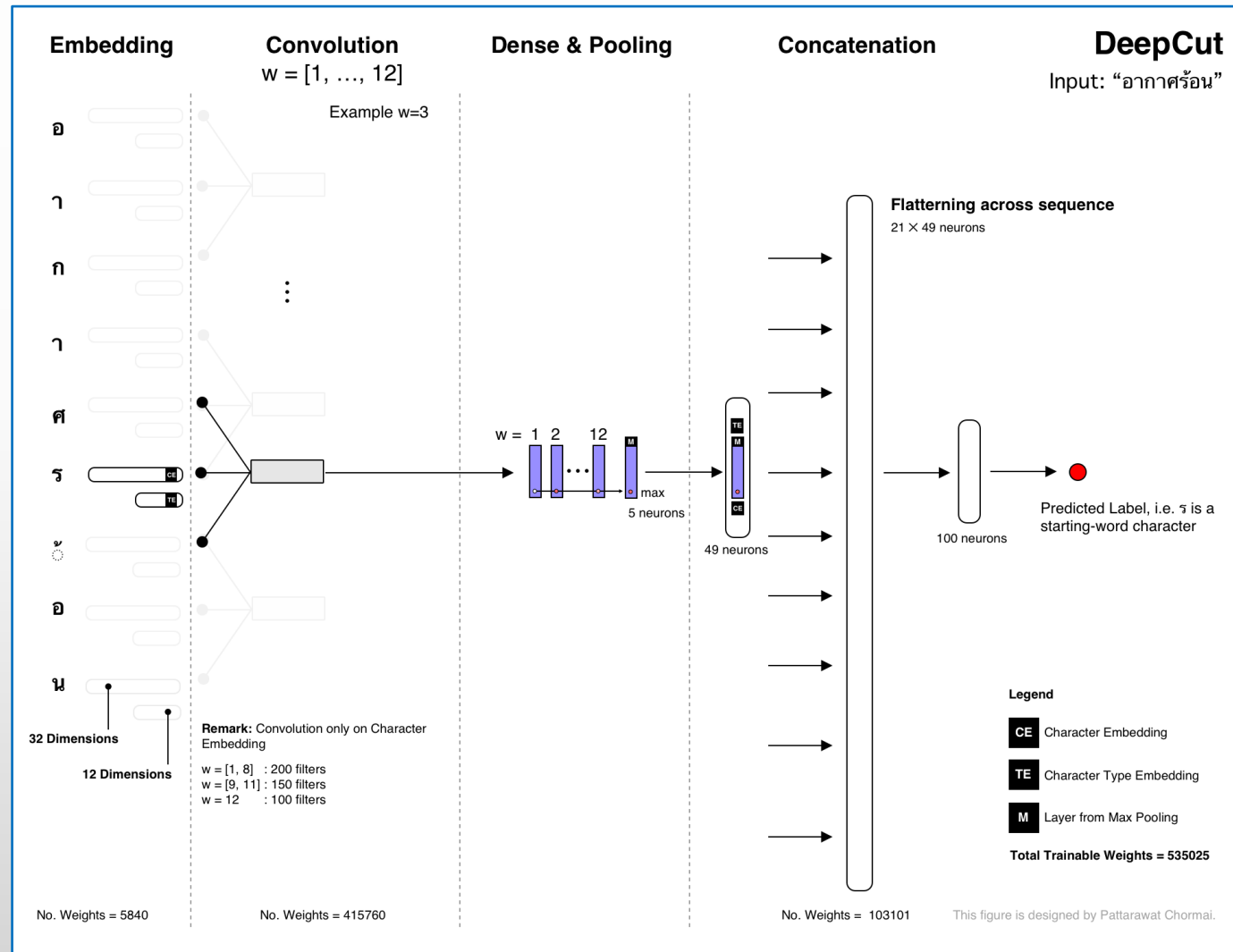
^aBest of the BEST 2009 Award Winner

^bBEST Student 2009 Award Winner

BEST 2009 : Thai word segmentation software contest

<https://ieeexplore.ieee.org/document/5340941>

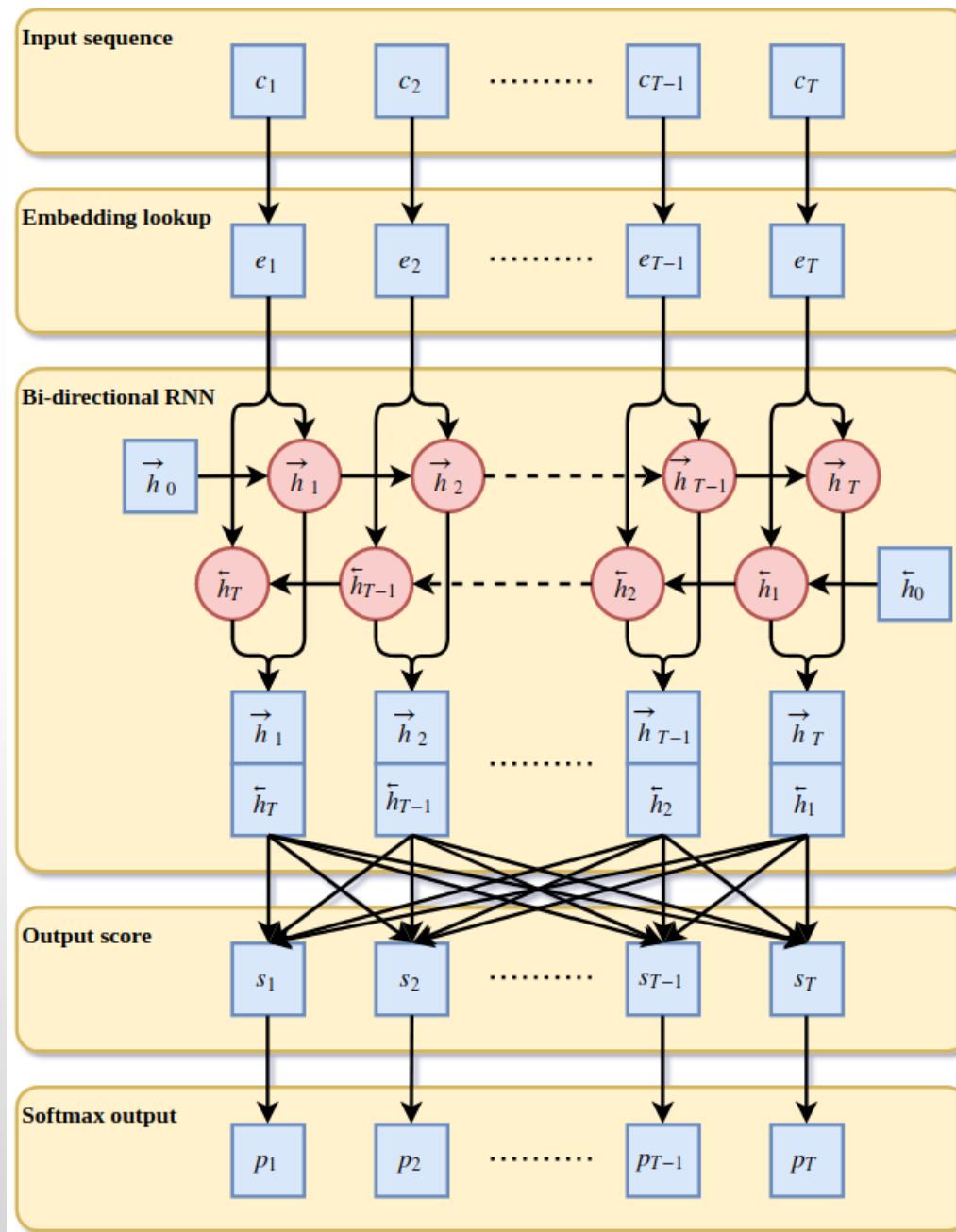
DeepCut (Thai-Tokenization)



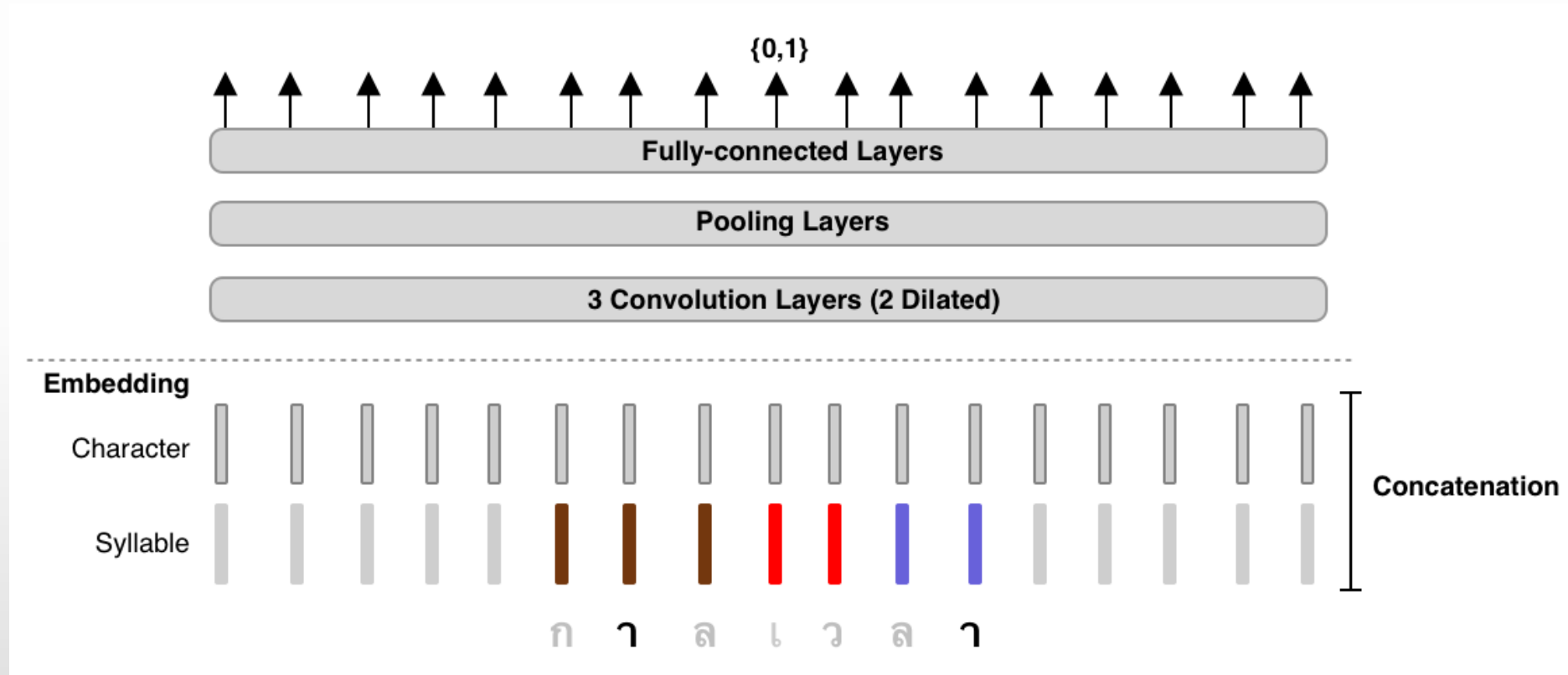
SERTIS (Thai-Tokenization)

Thai word segmentation with bi-directional RNN

<https://www.sertiscorp.com/th/november-20-2017>



AttaCut (Thai-Tokenization)



TL;DR: 3-Layer Dilated CNN on syllable and character features. It's **6x faster** than DeepCut (SOTA) while its WL-f1 on BEST is **91%**, only 2% lower.

Thai-Tokenization Quality

		Others			Ours	
Last Updated: 29/08/2019		PyThaiNLP newmm	Sertis Bi-GRU	DeepCut	<u>AttaCut-C</u>	<u>AttaCut-SC</u>
BEST Validation Set						
Character-Level	precision	0.94±0.11	0.95±0.10	0.99±0.05	0.97±0.07	0.98±0.05
	recall	0.83±0.09	0.99±0.02	0.99±0.03	0.98±0.04	0.99±0.03
	f1	0.88±0.08	0.97±0.07	0.99±0.04	0.98±0.05	0.99±0.04
Word-Level	precision	0.73±0.16	0.91±0.14	0.97±0.07	0.94±0.10	0.96±0.08
	recall	0.65±0.16	0.94±0.10	0.97±0.07	0.94±0.09	0.97±0.08
	f1	0.68±0.15	0.93±0.12	0.97±0.07	0.94±0.10	0.97±0.08
BEST Test Set						
Character-Level	precision	0.91±0.15	0.92±0.11	0.96±0.08	0.94±0.10	0.95±0.09
	recall	0.85±0.09	0.98±0.04	0.98±0.04	0.98±0.04	0.98±0.04
	f1	0.86±0.11	0.95±0.08	0.97±0.06	0.96±0.07	0.96±0.07
Word-Level	precision	0.70±0.19	0.85±0.18	0.92±0.14	0.88±0.17	0.91±0.15
	recall	0.64±0.18	0.90±0.14	0.93±0.12	0.91±0.14	0.92±0.13
	f1	0.67±0.19	0.87±0.16	0.93±0.13	0.89±0.16	0.91±0.14

Thai-Tokenization Quality

		Others			Ours	
Last Updated: 29/08/2019		PyThaiNLP newmm	Sertis Bi-GRU	DeepCut	<u>AttaCut-C</u>	<u>AttaCut-SC</u>
BEST						
C		Tokenizing Wisersight's Training Set (Statistics: 24k lines, 1.5M characters)				
	DeepCut			846.98±2.97 :: 1.0x		±0.05
	Sertis's Bi-GRU		309.71±1.54 :: 2.7x	522.56±2.78 :: 1.0x		±0.03
	AttaCut-SC		185.97±2.31 :: 2.8x			±0.04
	AttaCut-C		93.51±0.67 :: 9.1x			±0.08
	PyThaiNLP		93.78±0.23 :: 5.6x			±0.08
			59.19±0.09 :: 14.3x			
			52.71±0.51 :: 9.9x			±0.09
			7.43±0.09 :: 113.9x			±0.04
			7.29±0.05 :: 71.7x			±0.07
						±0.15
Word-Level	Recall	0.64±0.10	0.80±0.14	0.90±0.12	0.91±0.14	0.92±0.13
	f1	0.67±0.19	0.87±0.16	0.93±0.13	0.89±0.16	0.91±0.14

Word of caution

- A tokenizer trained on social text might not be able to cut simple words like

	WS160	TNHC
Deepcut	93.8	93.5
Attacut	93.5	80.8

<https://www.aclweb.org/anthology/2020.emnlp-main.315/>

- Examples

- มะม่วง มะละกอ ---> มะ | ม่วง | มะ | ละ | กอ
- หมูกรอบ ---> หมู | กรอบ
- ข้าวผัดคะน้าหมูกรอบหนึ่งจาน ---> ข้าว | ผัด | คะน้า | หมู | กรอบ | หนึ่ง | จาน

- Effect downstream task

Word of caution

- A tokenizer trained on social text might not be able to cut simple words like

	WS160	TNHC
Deepcut	93.8	93.5
Attacut	93.5	80.8

<https://www.aclweb.org/anthology/2020.emnlp-main.315/>

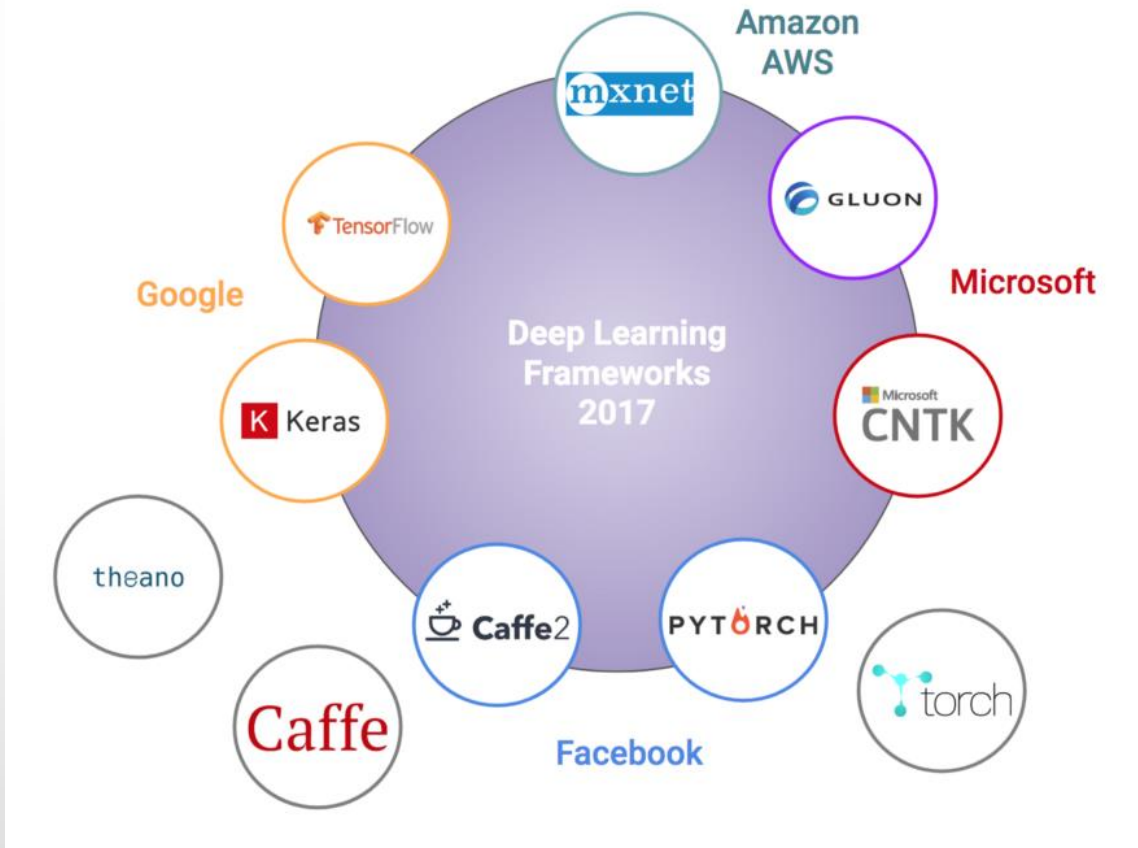
- Examples

- มะม่วง มะละกอ ---> มะ | ม่วง | มะ | ละ | กอ
- หมูกรอบ ---> หมู | กรอบ
- ข้าวผัดคะน้าหมูกรอบหนึ่งจาน ---> ข้าว | ผัด | คะน้า | หมู | กรอบ | หนึ่ง | จาน

**Might need
Dictionary-based !**

- Effect downstream task

NLP tools



<https://towardsdatascience.com/battle-of-the-deep-learning-frameworks-part-i-cff0e3841750>

NLP Libraries

NLTK Documentation

Search

Natural Language Toolkit

Natural Language Analysis with Python NLTK

NLTK Documentation
API Reference
Example Usage

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as Wordnet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.

<https://www.nltk.org/>

spaCy

Industrial-Strength Natural Language Processing

Get things done

Blazing fast

Awesome ecosystem

<https://spacy.io/>

Hugging Face

Search models, documentation

Course

Search documentation

MAIN

EN

Computer Vision

- Image Classification
- Image Segmentation
- Zero-Shot Image Classification
- Image-to-Image
- Unconditional Image Generation
- Object Detection
- Video Classification
- Depth Estimation

0. SETUP

1. TRANSFORMER MODELS

Introduction

Natural Language Processing

- Translation
- Fill-Mask
- Token Classification
- Sentence Similarity
- Question Answering
- Summarization
- Zero-Shot Classification
- Text Classification
- Text2Text Generation
- Text Generation
- Conversational
- Table Question Answering

Audio

- Automatic Speech Recognition
- Audio Classification
- Text-to-Speech
- Audio-to-Audio
- Voice Activity Detection

Multimodal

- Feature Extraction
- Text-to-Image
- Visual Question Answering
- Image-to-Text
- Document Question Answering

51

NLP Library for Thai

PyThaiNLP

We build Thai NLP.

Welcome to The Official PyThaiNLP Project Website.

The PyThaiNLP Project is a Thai Natural Language Processing project. We build softwares and datasets for Thai language. Our Main Project is PyThaiNLP.

See all our project at pythainlp.github.io/projects/

PyThaiNLP is a Python package for text processing and linguistic analysis, similar to nltk, with focus on Thai language.

<https://pythainlp.github.io/>

<https://pythainlp.github.io/docs/2.0/index.html>

PyThaiNLP documentation

PyThaiNLP is a Python library for Thai natural language processing (NLP).

Website: [PyThaiNLP.github.io](https://pythainlp.github.io)

Notes

- FAQ
- Command Line
- Getting Started
- Tutorial Notebooks
- Installation
- FAQ
- License

Package reference:

- pythainlp.segment
- pythainlp.benchmarks
- pythainlp.corpus
- pythainlp.generator
- pythainlp.parser
- pythainlp.morphology
- pythainlp.spell
- pythainlp.tokenize
- pythainlp.tag

AttaCut

A Fast and Accurate Neural Thai Word Segmenter

- Owner and Maintainer: Pattarawat Chomkiet
- Website: pythainlp.github.io/attacut
- GitHub: [PyThaiNLP/attacut](https://github.com/PyThaiNLP/attacut)

Docker Thai Tokenizers

This repository is a collection of almost all Thai tokenizers that are publicly available. Having this collection allows us to try each algorithm as ease via Docker.

- Owner and Maintainer: Pattarawat Chomkiet
- GitHub: [PyThaiNLP/docker-thai-tokenizers](https://github.com/PyThaiNLP/docker-thai-tokenizers)

Lao language

Lao language resources for PyThaiNLP

- Owner and Maintainer: Wongsongkh Phanthongphaban
- GitHub: [PyThaiNLP/lao-language](https://github.com/PyThaiNLP/lao-language)

lexicon-thai

Demo: Word tokenization in Thai

[https://drive.google.com/file/d/1oes0gZeddFbbsV4ymgDv4WyFDzreVqY9/view?usp=share link](https://drive.google.com/file/d/1oes0gZeddFbbsV4ymgDv4WyFDzreVqY9/view?usp=share_link)