NAIST Lecture (Nov. 12, 2025)

Shohei Higashiyama
Affiliate Assistant Professor@NAIST NLP Lab

NLP 2: Lexical Analysis
—Word Segmentation and Part-Of-Speech Tagging

Table of Contents:

Announcements for This Session

- **Materials**
  - Lecture materials are stored in the following folder:
    - https://drive.google.com/drive/folders/18gbP6ZwMhYC1QJwO5th29cw7tYsHR97f?usp=sharing
  - You can find this URL from the syllabus (https://edu-portal.naist.jp/)

- **Assignments**
  - This session requires you to submit a report (some simple quizzes related to lexical analysis and tokenization).
  - Assignment details will be announced at the end of the lecture.

- **Questions**
  - If you have any questions, please submit them through Slido:
    - https://app.sli.do/event/gdPg4DdX4RN1jaiNBhv6fh

Slido link for Q&A

Part 1:
Fundamental Concepts and Background

Task

- What is a *task* in NLP?
  - A problem that a system or model needs to solve.
  - Defined by its input and output.

Example: Recognizing Textual Entailment (RTE)

A pair of two texts

[Premise] She bought a new car.

[Hypothesis] She has a car.

One of three labels:
"entailment," "non-entailment," or "neutral"

entailment

Example: Machine Translation (MT)

Text in the source language

体調はどうですか？

taichō wa dō desuka

Text in the target language

How are you feeling?

Representative Tasks in Lexical Analysis

- Tokenization
- Word Segmentation
- Part-of-Speech Tagging
- Morphological Analysis

[Notes]

- The term *Lexical Analysis* is sometimes used to encompass various lexical-level linguistic analysis tasks, but it is not as well-established as individual task names.
- This lecture regard tokenization as a lexical analysis-related task for convenience, but it not usually considered as such, as it does not necessarily identify linguistic units.

Tokenization and Word Segmentation

- **Token/Tokenization**
  - A *token* refers to a unit of text for processing in NLP.
  - Classical tokenizers (like NLTK) split a sentence into words and punctuation marks as tokens.

Sentence: Mr. Smith isn’t worried. ▶ Sequence of tokens: | Mr. | Smith | is | n’t | worried | . |

- **Word Segmentation (=a type of tokenization)**
  - A task of dividing an *unsegmented sentence* into words.
  - Typically performed for *unsegmented languages* w/o spaces, such as Japanese and Chinese.

Sentence: 言語は複雑だ ▶ Sequence of words: | 言語 | は | 複雑だ |

Word boundaries

言語	は	複雑だ
gengo	*wa*	*fukuzatsu-da*
languages	are	complex

* “は” is actually a topic marker.

Part-of-Speech Tagging

- Part-of-Speech (POS)
  - A grammatical category of a word, such as *noun* and *verb*, which indicates how it functions in a sentence.
  - Helps an NLP system understand the roles of words within a sentence.

- POS Tagging
  - A task of assigning POS tags for words in a sentence.

Sequence of words

["Mr.", "Smith", "is", "n't", "worried."]

▼

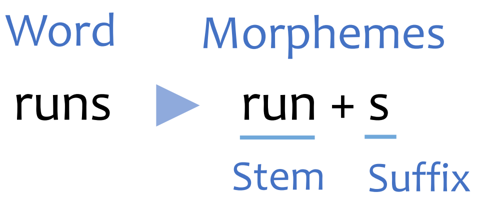
Sequence of POS tags

["PROPN", "PROPN", "AUX", "PART", "ADJ", "PUNCT"]

Universal POS tags (Universal Dependencies Project)

ADJ: adjective
ADP: adposition
ADV: adverb
AUX: auxiliary
CCONJ: coordinating conjunction
DET: determiner
INTJ: interjection
NOUN: noun
NUM: numeral
PART: particle
PRON: pronoun
PROPN: proper noun
PUNCT: punctuation
SCONJ: subordinating conjunction
SYM: symbol
VERB: verb
X: other

Morphological Analysis

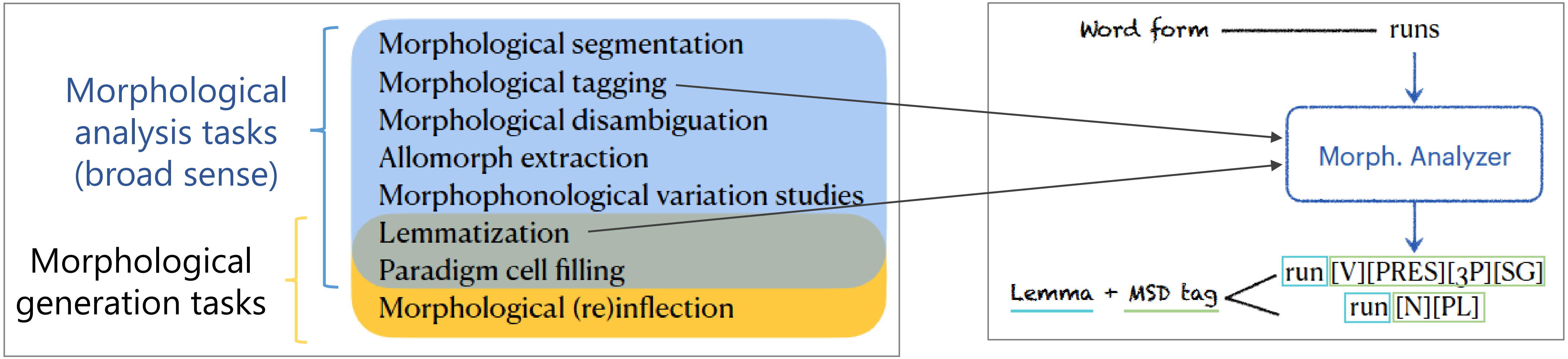


* The *lemma* and *root* are also "run."

- Morpheme: The smallest linguistic unit that carries meaning.
- Definition of "Morphological Analysis"
  - Broad sense: A general term for analysis-focused morphological learning tasks
  - Narrow sense: A specific term refers to the combination of *lemmatization* and *morphological tagging*

Morphological learning tasks

Morphological analysis (narrow sense)



Cited From [Liu '21]

* MSD: Morphosyntactic description
* MSD tags can be regarded as fine-grained POS tags.

Japanese Morphological Analysis

- Definition
  - Typically refers to a complex sentence-level task, involving word segmentation, POS tagging, and lemmatization (inflection processing).

Input sentence

昨日は楽しかった。

kinō wa tanoshikatta

I had a good time yesterday.

▶

Output

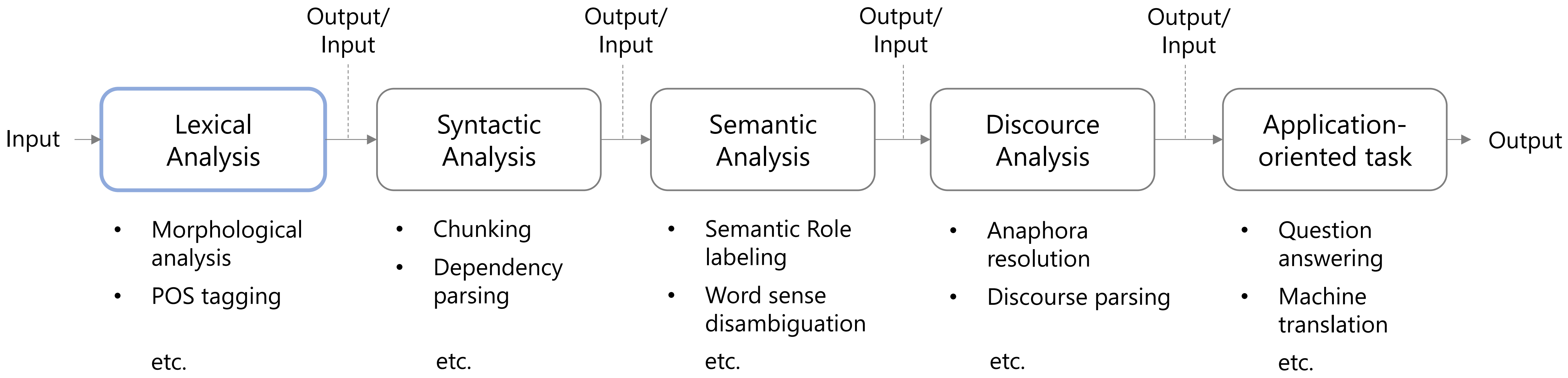
Word:	昨日	は	楽しかっ	た
Lemma:	昨日	は	楽しい	た
POS tag:	Noun	Particle	Verb	Suffix

[Notes]

- Each “word” token actually correspond to a word, morpheme, or an intermediate unit. The segmentation granularity depends on the segmentation criterion (POS tag system).
- Thus, in Japanese NLP, *morhpmes* and *words* are usually not strictly distinguished and are often used interchangeably.

Pipelined Task Flow in Traditional NLP

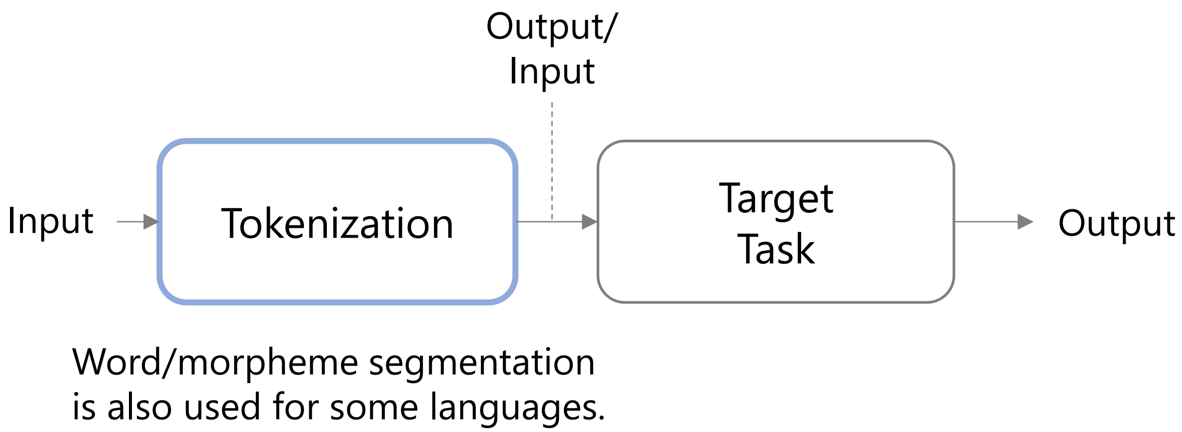
- In traditional NLP, lexical analysis tasks <u>were</u> essential preprocessing steps.



* Each application-oriented task does not necessarily require all the preceding lower-level tasks.

End-to-End Task Flow in Current NLP

- In current neural NLP,
  tokenization is usually the only mandatory prerequisite step for a target task.



Other Fundamental Concepts

- Vocabulary
  - A set of words/tokens collected based on specific criteria.
  - An NLP model has a vocabulary of tokens that the model can process.

- Type vs. Token
  - Word type: Unique form of a word in a text/vocabulary.
  - Word token: Each instance of a word appeared in a text.

This sentence has
5 word tokens & 4 word types.

"a person has a pen"

- Ambiguity
  - The possibility of multiple interpretations.
  - *Word segmentation ambiguity* can impact the performance of downstream tasks.

米原発熱海行き *Maibara-hatsu Atami-yuki*	米原|発熱|海|行き	For Maibara Heat Sea
	米|原発|熱海|行き	U.S. Nuclear Power Plant to Atami
	米原|発|熱海|行き	From Maibara to Atami **Appropriate**

- These are machine translation outputs that I obtained on a previous day.
- 米原 (Maibara) and 熱海 (Atami) are place names, so the third one is natural.

* NLP faces many challenges due to various kinds of ambiguities in languages!

Other Fundamental Concepts

- **Vocabulary**
  - A set of words/tokens collected based on specific criteria.
  - An NLP model has a vocabulary of tokens that the model can process.

- **Type vs. Token**
  - Word type: Unique form of a word in a text/vocabulary.
  - Word token: Each instance of a word appeared in a text.

This sentence has
5 word tokens & 4 word types.

"a person has a pen"

- **Ambiguity**
  - The possibility of multiple interpretations.
  - *Word segmentation ambiguity* can impact the performance of downstream tasks.

米原発熱海行き
Maibara-hatsu Atami-yuki

米原|発熱|海|行き → For Maibara Heat Sea
米|原発|熱海|行き → U.S. Nuclear Power Plant to Atami
米原|発|熱海|行き → From Maibara to Atami **Appropriate**

- These are machine translation outputs that I obtained on a previous day.
- 米原 (Maibara) and 熱海 (Atami) are place names, so the third one is natural.

* NLP faces many challenges due to various kinds of ambiguities in languages!

Summary of Part 1

- We have looked at:
  - Tokenization as a prerequisite step.
  - Definitions of lexical analysis tasks:
    word segmentation, POS tagging, and morphological analysis.
  - Fundamental concepts: token, vocabulary, and word tokens/types.
  - The role of tokenization and lexical analysis in the traditional/current NLP task flows.

Somewhat Focusing on
Word Segmentation

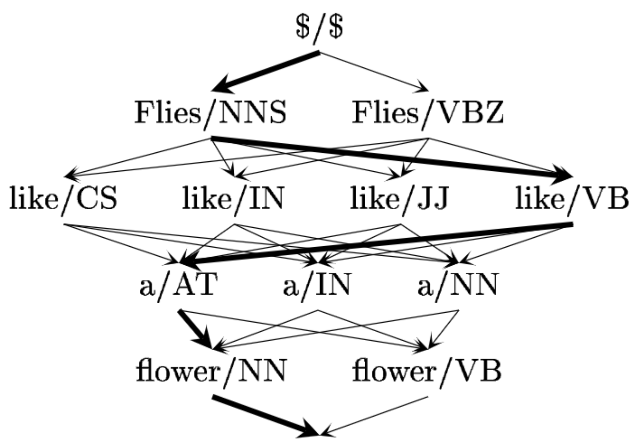
Part 2:
Some Methods for Lexical Analysis Tasks

Progression of Methods for Word Segmentation and POS Tagging

- From around 2000 to the mid-2010s:
  - Statistical machine learning models were primarily used.

    cf. https://aclweb.org/aclwiki/POS_Tagging_(State_of_the_art)

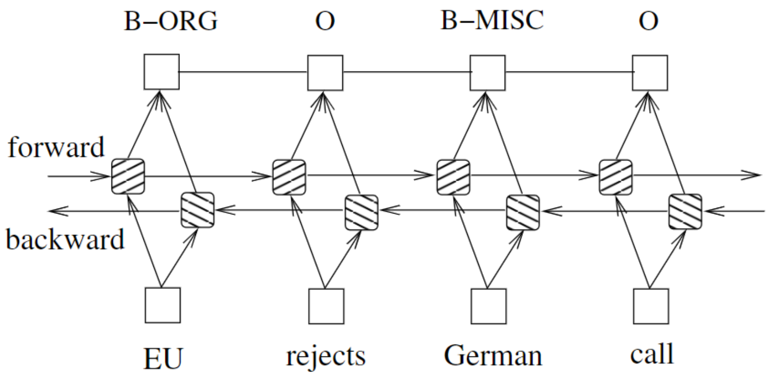
    [Liu+ '23] Survey on Chinese Word Segmentation



A lattice structure for the HMM-based POS tagger [Lee+ '00]

- From the mid-2010s to the early 2020s:
  - Neural network models were actively developed and have achieved substantial performance improvements (particularly in Chinese word segmentation).

Note: Statistical models, such as MeCab and Jieba, remain popular as practical tools.



BiLSTM-CRF Tagger [Huang+ '15]
(Illustated outputs are labels for an NER task.)

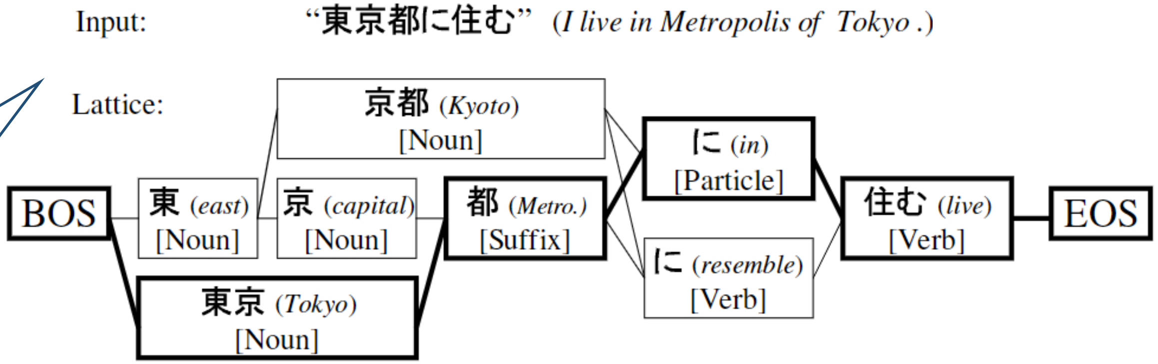
Statistical Methods for Japanese Morphological Analysis (JMA)

- Characteristics of statistical methods
  - Typical methods are *lattice*-based approaches relying on dictionaries.
  - Computationally efficient and fast.
  - Available off-the-shelf models can achieve high accuracy for well-formed text (e.g., news).

    * It is because such models are trained with texts like news articles.

- MeCab [Kudo+ '04]
  - One of the de facto standard tools for JMA.
  - Based on Conditional Random Fields.

1. Construct the *lattice* for each input sentence while referring to the MA dictionary.
2. Search the best path using the *Viterbi algorithm* based on the trained model parameters.



A **word lattice** is a graph in which nodes represent words, and edges indicate that the two nodes can be connected.

Neural Methods for Word Segmentation

- Characteristics of neural methods
  - Often treat word segmentation as *sequence labeling*.
  - <u>Language-independent</u> and easily extensible to a multi-task model.
  - Can obtain benefit from powerful *pretrained language models* like BERT.

Sequence labeling:
a type of a task that predicts a label sequence for an input token sequence

Input character tokens:

[奈, 良, に, は, 鹿, が, い, る]

Nara ni-wa shika-ga iru

There are deer in Nara.

▼

Label sequence to be predicted:

[B, E, B, B, B, B, B, E]

奈良|に|は|鹿|が|いる

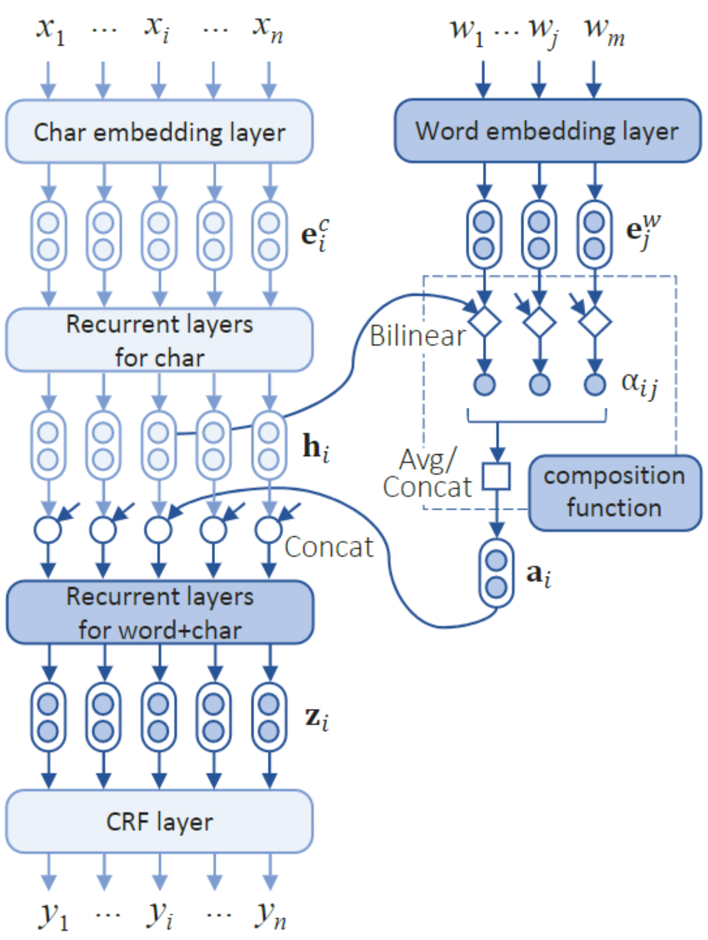
B: Beginning of a word
E: End of a word

* Popular tag schemas:
BE (=0/1), BIE, BIES, etc.

* Many NLP tasks other than word segmentation can also be formulated as sequence labeling.

Neural Methods for Word Segmentation

- A character-word hybrid model [Higashiyama+ '19]
  - Relied a BiLSTM-CRF, which was the de facto standard architecture for neural sequence labeling (before the BERT era).
  - Achieved state-of-the-art accuracy for both Japanese and Chinese word segmentation.
    - By changing the training data, the same model can be used for different languages.



Neural Methods for Word Segmentation

- Model comparison on various Japanese texts
  - MeCab achieved high accuracy for GEN (general) domain.
    - The model was trained on GEN domain data.
  - BERT achieved high accuracy for many domains.
    - The model was pretrained on Japanese Wikipedia texts and fine-tuned on GEN domain data.

* BERT [Devlin+ '19] is a powerful neural NLP model and a prominent example of a masked language model (MLM).

Neural methods (based on pretrained language models) have the advantage of robustness to domain shift.

Acucracy (F1 scores) of Japanese word segmenters on various domains ("dom.") [Higashiyama+ '22]

Dom.	Unknown Tok/Type Ratio	MeCab D_s		BL-LWP D_s, D_t, U_t		BERT –	
		Seg	POS	Seg	POS	Seg	POS
GEN	2.7 / 16.1	99.6	99.0	98.9	98.3	99.4	99.1
ENE	2.5 / 15.4	99.3	98.9	99.6	99.2	99.7	99.4
TRA	3.0 / 18.2	98.8	98.4	99.4	98.9	99.6	99.2
ENV	3.2 / 15.1	98.8	98.1	99.3	98.7	99.5	99.2
MAN	3.3 / 19.5	98.6	98.2	99.4	99.0	99.6	99.3
CON	3.5 / 19.5	98.9	98.1	99.2	98.6	99.5	99.1
AGR	4.5 / 21.0	98.5	98.0	99.0	98.4	99.4	99.0
THM	4.5 / 24.0	98.4	97.7	99.1	98.3	99.4	98.8
INF	4.7 / 22.6	97.9	97.5	99.1	98.5	99.5	99.1
MEC	5.0 / 25.3	98.4	97.8	99.3	98.7	99.5	99.1
NUC	5.3 / 20.2	98.1	97.3	98.9	98.0	99.4	98.9
CHE-I	5.5 / 23.7	97.9	97.3	99.0	98.3	99.5	99.0
ETH	5.5 / 24.5	98.5	97.8	99.3	98.4	99.4	98.8
MED	5.6 / 27.0	97.1	96.6	99.1	98.6	99.5	99.1
SYS	5.6 / 24.8	98.4	97.7	98.9	98.0	99.4	98.7
ELC	5.8 / 29.4	97.4	97.0	99.0	98.5	99.5	99.1
PAT	6.0 / 26.8	97.0	96.8	99.1	98.7	99.4	99.2
CHE-E	6.1 / 23.7	97.9	97.0	99.0	98.0	99.2	98.7
MIN	6.6 / 22.6	98.0	97.4	98.8	98.1	99.0	98.6
BIO	6.7 / 30.2	96.7	96.0	98.8	98.0	99.3	98.7
PHY	7.5 / 29.6	97.1	96.4	98.5	97.7	99.2	98.8
CHE-B	8.1 / 35.4	97.0	96.1	98.5	97.4	99.1	98.4
EMR	11.2 / 30.2	95.4	91.9	95.6	92.5	97.1	94.0
DIE	0.9 / 7.5	98.0	97.6	97.7	97.0	97.9	97.4
LAW	2.1 / 11.0	97.4	97.0	97.6	97.4	97.9	97.8
PRM	2.8 / 11.1	98.7	98.1	97.7	96.8	98.3	97.8
TBK	4.4 / 19.0	99.0	97.0	97.7	95.5	98.7	96.8
VRS	19.7 / 47.4	87.3	82.3	81.8	75.1	87.1	83.0

ENE to EMR: Scientific documents.
DIE to PRM: Government documents.
TBK to VRS: Other documents.

Practical Systems for Japanese Word Segmentation

- Statistical methods, particularly MeCab, are still widely used for high processing speed.
- Juman++ V2 [Tolmachev+ '20] utilizes a recurrent neural network (RNN) language model to achieve high accuracy while maintaining practical processing speed.

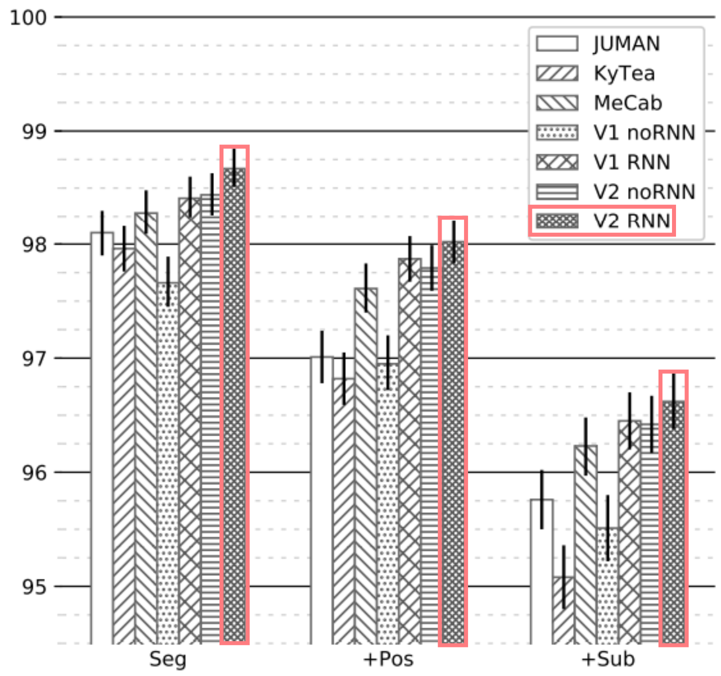
Processing speed fo Japanese word segmenters

	Analyzer	Speed (sents/s)	Ratio
Rule-based method	JUMAN	8,802	1.00
Statistical method	MeCab	52,410	0.17
	KyTea (Jumandic)	4,892	1.79
	KyTea (Unidic)	1,995	4.41
Juman++: Statistical/Neural hybrid method	V1 noRNN	27	328.82
	V1 RNN	16	535.72
	V2 noRNN	7,422	1.18
	V2 RNN	4,803	1.83

* Results are from [Tolmachev+ '20].

* Vaporetto and vibrato, implementations of KyTea and MeCab like tokenizers, achieve faster tokenization than them.

Accuracy (F1 scores) of Japanese word segmenters on web text (KWDLC data)



Summary of Part 2

- We have looked at:
  - Progression of lexical analysis methods from statistical to neural models.
  - Examples and characteristics of statistical and neural methods (for Japanese morphological analysis and word segmentation):
    - Optimized statistical models are efficient and fast, whereas neural models are language-independent and have the potential to achieve robust accuracy across domains.
  - Importance of processing speed: practical systems pursue both speed and accuracy.

Part 3:
Tokenization in the Neural NLP Era

What are Suitable Token Units for Various NLP Tasks?

- Word Segmentation: *Character*
  - Words cannot be used before predicting them.

Example inputs/outputs

Input character tokens:

[奈, 良, に, は, 鹿, が, い, る]

Nara ni-wa shika-ga iru

There are deer in Nara.

▼

Label sequence to be predicted:

[B, E, B, B, B, B, B, E]

B: Beginning of a word

E: End of a word

What are Suitable Token Units for Various NLP Tasks?

- Most other tasks: *Word* (?)
  - Using words seems efficient and easy.
  - Using characters leads to long token sequences, which increases *computation costs* and raises *modeling difficulty*.

Example inputs/outputs for Sentiment Analysis

When using character tokens:

[I, _, d, o, _, n, o, t, _, l, i, k, e, _ , t, h, i, s, _, m, o, v, i, e]

When using word tokens:

[I, do, not, like, this, movie]

Label to be predicted:

negative

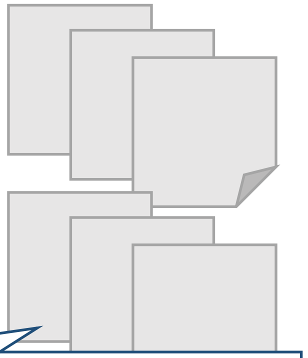
A system is required to grasp the information expressed by the combination of these tokens.

* Previous character-level models have shown lower accuracy than (sub)word-level models [Al-Rfou+ ‘19]. However, it has been demonstrated that Transformers with deep layers and a large number of parameters, even at the byte level, can achieve performance competitive with (sub)word-level models [Choe+ ‘19].

Problems of Using Word Tokens in Downstream Tasks

- A large vocabulary (with size |V|) leads to:
  - Expensive (infeasible) computation, particularly for language generation models
  - A large number of model parameters

Corpus (Set of texts)



A large corpus may include millions of word types.

Words appeared in the corpus

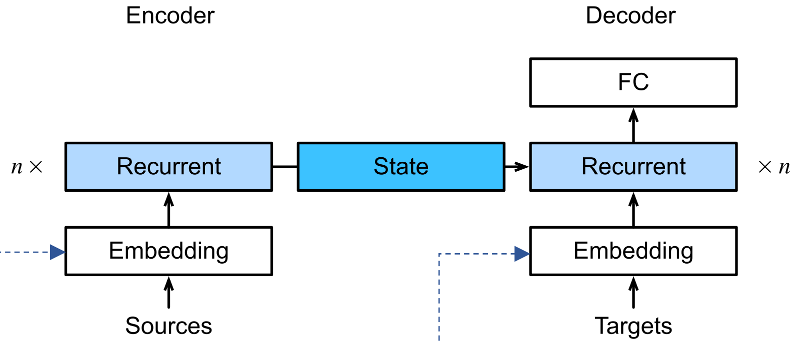
Model's vocabulary

ID	Token
0	[UNK]
1	the
2	is
…	…
50000	placid
…	…

Neural network model

RNN with attention

(Cited from Dive into Deep Learning)



When |V|=32,000 and dim=512, word embedding parameters account for 25% of GRU model.

To keep the vocabulary size practical (typically <100K), a special unknown token is used to represent all words outside the vocabulary, but this is not ideal.

A Solution: Subword

- Subword tokenization
  - Breaks down less common words into *subword* tokens based on statistical criteria.

Input text: There have been significant advancements in NLP technologies.

Output tokens:

["there", "have", "been", "significant", "advancement", "##s", "in", "nl", "##p", "technologies", "."]

The symbol "##" represents non-initial token in a word

(tokenizer: `google-bert/bert-base-multilingual-uncased`)

- Pros:
  - Vocabulary size is controllable as a model's hyper-parameter.
  - Unknown words are rare: Most words can be represented by combinations of subwords.

These drawbacks are not often practically critical, and subwords have become the de facto standard.

- Cons:
  - Subwords do not align with the boundaries of linguistically meaningful units (e.g., morphemes).
  - A tokenizer depends on specific training data, leading to less portability.

    * word segmenters also have this limitation.

Subword Tokenization Algorithms

- Three popular algorithms:
  - WordPiece [Schuster+ '12]
  - Byte Pair Encoding (BPE) [Sennrich+ '16]
  - Unigram Language Model (LM) [Kudo '18]

  Explained in detail in later slides.

  cf. https://huggingface.co/docs/transformers/tokenizer_summary.

- Two phases for subword tokenization
  - Train a tokenization model from a training corpus.
  - Use the model to tokenize new text.

Byte Pair Encoding (BPE)

- Training algorithm
  1. Create an initial small subword vocabulary of all characters in the training corpus.
  2. Merge the two consecutive subwords that occur most frequently in the corpus.
     - The tokenizer learns the merge rule.
  3. Repeat step 2. until the vocabulary reaches the desired size.

Indicates that “hugs” occurs five times, and so on.

Corpus: ("hug", 10), ("pug", 5), ("pun", 12), ("bun", 4), ("hugs", 5)

▼

Merge “u” and “g” (frequency: 20)

Vocabulary: ["b", "g", "h", "n", "p", "s", "u"]

Corpus (subword-based): ("h" "u" "g", 10), ("p" "u" "g", 5), ("p" "u" "n", 12), ("b" "u" "n", 4), ("h" "u" "g" "s", 5)

▼

New subword

Merge “u” and “n” (frequency: 16)

Vocabulary: ["b", "g", "h", "n", "p", "s", "u", "ug"]

Corpus (subword-based): ("h" "ug", 10), ("p" "ug", 5), ("p" "u" "n", 12), ("b" "u" "n", 4), ("h" "ug" "s", 5)

...

Example is from Hugging Face

Unigram Language Model

Likelihood decrease or *loss* for s_m

$$\boxed{\mathcal{L}(D,V) - \mathcal{L}(D, V \setminus \{s_m\})}$$

- Training algorithm
  1. Create an initial <u>large</u> subword vocabulary V consisting of a reasonably large number of possible subwords from the training corpus $D = \{x^{(i)}\}$.
  2. Calculate the <u>likelihood decrease for each subword</u> $s_m \in$ V when it is removed, and <u>Remove</u> the top k (e.g., 20) percent of subwords that most decrease the likelihood.
     – Single-character subwords are retained regardless of their effect on the loss to avoid the out-of-vocabulary problem.
  3. Repeat step 2. until the vocabulary reaches the desired size.

All sentences $D = \{x^{(i)}\}$

$$\underbrace{\mathcal{L}(D,V)}_{\text{Likelihood over } D} = \sum_{i=1}^{|D|} \log \left(\overbrace{\sum_{\underbrace{y \in S(x^{(i)}, V)}_{\text{Set of all possible segmentation } y \text{ based on the vocab V}}} \left(\underbrace{\prod_{j=1}^{|y|} p(y_j)}_{\text{Probability } P(y) \text{ of segmentation } y = (y_1, \ldots, y_{|y|})} \right)}^{P(S(x^{(i)}), \mathrm{V})} \right)$$

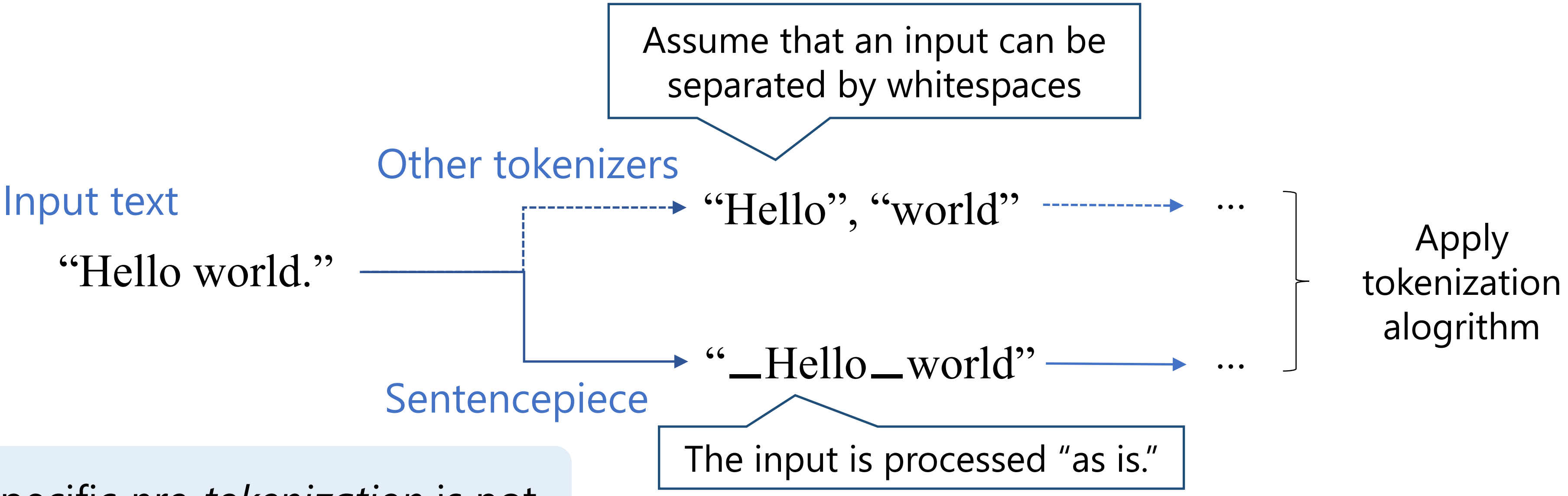
Sum

y = "_Hell/o/_world"	$P(y)$
y = "_H/ello/_world"	$P(y)$
y = "_He/llo/_world"	$P(y)$
y = "_/He/l/l/o/_world"	$P(y)$
y = "_H/el/l/o/_/world"	$P(y)$
…	

x_i = "Hello world"

Pre-tokenization-free Tokenizer

- Sentencepiece [Kudo+ '18]
  - Subword tokenization tool (¬ algorithm) that implemented BPE and Unigram LM.
  - Enabled language-independent tokenization
    by treating all characters, including whitespace, as usual symbols.



Language-specific *pre-tokenization* is not required for unsegmented languages.

* The beginning of sentence and whitespaces are replaced by a meta character "▁"

Summary of Part 3

- We have looked at:
  - Suitable token unit: It depends on tasks, but word-level tokenization provides computation efficiency compared to character-level tokenization.
  - A clear drawback of word-level tokenization:
    It limits vocabulary size, and thus infrequent words are treated as unknown tokens to avoid impractical computational costs.
  - Subword tokenization as a solution:
    This has some (not critical) drawbacks, but has become the de facto standard.
  - Well-known subword tokenization algorithms: BPE and Unigram LM.
  - Sentencepiece: A truly language-independent tokenization tool,
    particularly useful for unsegmented languages.

Somewhat Focusing on Word Segmentation

Part 4: The Usage of Foundational Lexical Analysis in Current NLP

How Lexical Analysis is Currently Used: A Case of Japanese NLP

- Direct subword tokenization from raw text performs well in downstream tasks.
- Two-step tokenization is also often adopted.

Named entities: Specific real-world objects/concepts

Train line name | Anime title | Place name

Input: 今日は京王線で天気の子の聖地新宿へ

kyō wa keiō-sen de tenki-no-ko no seichi shinjuku e

Today I'm heading to Shinjuku, a location featured in Weathering with You, via the Keio Line.

- bert-japanese's tokenizer: Unigram LM
  - ▁|今日|は|京王|線で|天気|の|子の|聖地|新宿|へ
- llm-jp-3-1.8b-instruct's tokenizer: Unigram LM with MeCab pre-tokenization
  - ▁今日は|京王|線|で|天気|の|子|の|聖地|新宿|へ

Direct subword tokenization

Token boundaries do not align with word boundaries, particularly for named entities.

Two-step tokenization

First perform word segmentation and then build a subword vocabulary based on obtained words.

- This reduces cases of boundary conflicts.

* "▁" is a symbol representing the beginning of a non-spaced sequence.

* A recommended method of combining Sentencepiece and MeCab is explained here (in Japanese).

How Downstream Task Accuracy Differ by Tokenizers?

- It depends on tasks, but using words often produces good results (in Japanese NLP).
  - Direct subword tokenization led to a large performance drop for Named Entity Recognition.

Two-step: Use word segmenter as pre-tokenizer

Direct: Subword tokenizer only

Tokenizer Subword	Morphological	MARC-ja Accuracy	JSTS Spearman	JNLI Accuracy	JSQuAD F1	JCQA Acc	NER F1	UD LAS	Avg.
bert-base-japanese		95.5±0.1	85.3±0.3	86.8±0.6	86.4±0.2	76.6±0.8	85.6±0.2	93.3±0.1	87.1
BPE ($\mathcal{B}$)	Ⓜ MeCab	95.4±0.2	84.2±0.1	88.0±0.4	90.1±0.3	74.1±0.7	83.7±0.8	93.6±0.1	87.0
	Ⓙ Juman++	95.5±0.1	84.6±0.4	87.6±0.4	90.1±0.2	73.8±0.3	85.1±0.6	93.6±0.1	87.2
	Ⓢ Sudachi	95.5±0.1	84.2±0.2	88.2±0.3	90.2±0.2	74.2±0.6	83.5±0.6	93.8±0.1	87.1
	Ⓥ Vaporetto	95.6±0.1	84.8±0.2	87.5±0.3	89.9±0.2	74.2±1.1	84.1±0.9	93.7±0.1	87.1
	Nothing	95.4±0.2	82.8±0.2	87.2±0.2	88.7±0.3	72.8±0.8	62.9±1.1	93.4±0.1	83.3
WordPiece ($\mathcal{W}$)	MeCab	95.5±0.1	82.4±0.5	87.5±0.3	89.2±0.3	69.8±0.7	84.0±0.9	93.6±0.1	86.0
	Juman++	95.3±0.3	83.3±0.3	87.7±0.2	89.8±0.3	71.1±0.6	84.7±0.5	93.6±0.1	86.5
	Sudachi	95.3±0.2	83.7±0.3	87.2±0.4	89.6±0.1	70.0±0.9	82.4±0.6	94.0±0.1	86.0
	Vaporetto	95.3±0.2	83.6±0.1	88.0±0.4	89.7±0.2	71.0±0.4	84.0±0.8	93.8±0.1	86.5
	Nothing	85.5±0.0	N/A	55.3±0.0	10.1±0.1	20.0±0.8	0.0±0.0	63.8±0.9	33.5
Unigram ($\mathcal{U}$)	MeCab	95.4±0.3	84.6±0.4	88.3±0.4	89.5±0.3	74.5±0.8	83.1±1.0	93.4±0.2	87.0
	Juman++	95.4±0.2	84.3±0.3	87.8±0.3	89.9±0.2	74.9±1.2	84.1±0.4	93.4±0.1	87.1
	Sudachi	95.6±0.2	84.8±0.5	88.4±0.3	89.9±0.1	74.5±0.6	83.0±1.3	93.7±0.1	87.1
	Vaporetto	95.5±0.3	84.6±0.2	87.9±0.3	89.9±0.1	74.3±0.8	84.1±0.4	93.7±0.1	87.1
	Nothing	95.4±0.4	83.9±0.3	87.7±0.8	89.3±0.1	74.6±0.4	76.9±1.0	93.2±0.2	85.9

MARC-ja: Sentiment analysis; JSTS: Sentence similarity; JNLI: Entailment recognition; JSQuAD, JCQA: Question answering; NER: Named entity recognition; UD: Dependency parsing

Unigram achieved robust accuracy w/o word segmentation (except for NER)

* Results are from [Fujii+ '23].

A Case of a Morphologically Rich Language: Kinyarwanda

- KinyaBERT
  - Uses two-tier BERT architecture (morpheme and sentence-level encoders) designed specifically for this language.
  - Achieved high accuracy across tasks.

Words and morphemes of Kinyarwanda

Word	**Morphemes**
twagezeyo *'we arrived there'*	**tu . a . ger . ye . yo**
ndabyizeye *'I hope so'*	**n . ra . bi . izer . ye**
umwarimu *'teacher'*	**u . mu . arimu**

Models with BPE or morpheme tokenization yielded lower accuracy than KinyaBERT on most tasks.

Model	MRPC	QNLI	RTE	SST-2	STS-B	WNLI	NER	NEWS
XLM-R	82.6/76.0±0.6/0.6	78.1±0.3	56.4±3.2	76.3±0.4	69.5/68.9±1.0/1.1	63.7±3.9	71.8±1.5	84.0±0.2
BERT_{BPE}	82.8/76.2±0.6/0.8	81.1±0.3	55.6±2.8	79.1±0.4	68.9/67.8±1.8/1.7	63.4±4.1	74.8±0.8	**88.3**±0.3
BERT_{MORPHO}	82.7/75.4±0.8/1.3	80.8±0.4	56.7±1.0	80.7±0.5	68.9/67.8±1.5/1.3	65.0±0.3	72.8±0.9	86.9±0.3
KinyaBERT_{ADR}	84.4/**78.7**±0.5/0.6	81.2±0.3	58.1±1.1	80.9±0.5	73.2/72.0±0.4/0.3	65.1±0.0	**77.2**±1.0	88.0±0.3
KinyaBERT_{ASC}	**84.6**/78.4±0.2/0.3	**82.2**±0.6	**58.8**±0.7	**81.4**±0.6	**74.5/73.5**±0.2/0.2	65.0±0.2	76.3±0.5	88.0±0.2

Cited from [Nzeyimana+ '22]

* The results on the test data.

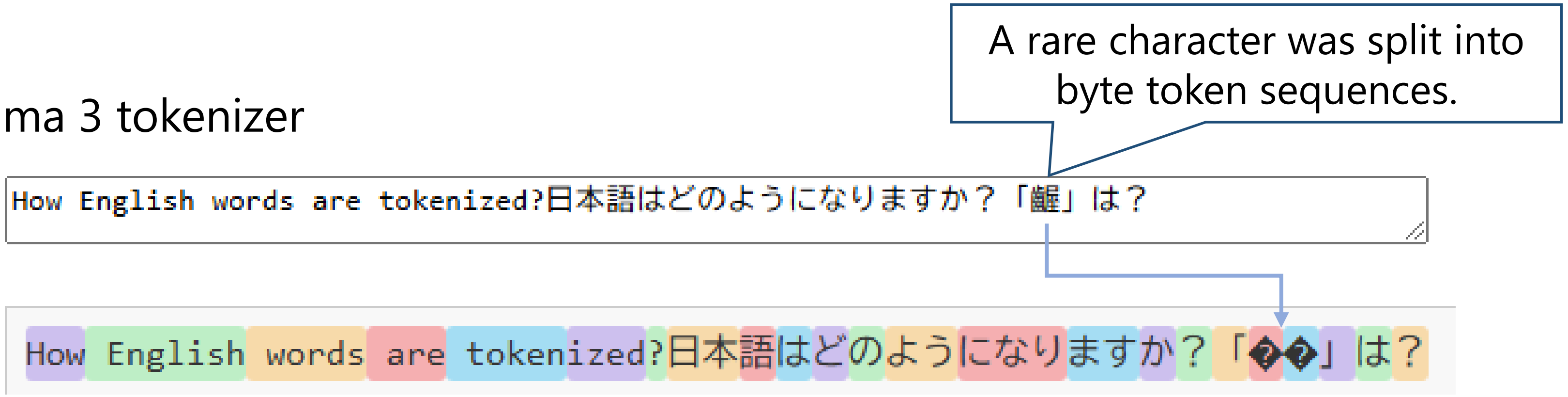
What Tokenization is Used in State-of-the-Art LLMs?

- Byte-level BPE
  - The vocabulary is initialized with 256 UTF-8 byte tokens, and new tokens are added by merging existing tokens.
  - Completely eliminates the issue of unknown words/characters.

* In UTF-8 encoding, each byte (=8 bits) can represent 2^8=256 possible values.

Adopted in:
- English-centric LLMs: OpenAI's GPT series, Llama 3, etc.
- Chinese-centric LLMs: Alibaba's Qwen series, etc.

Llama 3 tokenizer



https://belladoreai.github.io/llama3-tokenizer-js/example-demo/build/

What Tokenization is Used in State-of-the-Art LLMs?

- Two-step tokenization (word and subword)
  - Japanese-centric LLM
    - LLM-jp: MeCab+JumanDIC → Unigram LM
  - Japanese-centric LLM extended from English-centric LLM
    - Swallow: MeCab+UniDic → BPE

Efficient tokenization is important to minimize costs.

Input: 日本語の自然言語処理

nihon go-no shizen gengo shori

Tokenizers not optimized for non-Latin characters increase training/inference costs for non-Latin languages.

➢ Before vocabulary expansion (Base model: Llama 2)

日|本|語|の|自|然|言|語|<0xE5>|<0x87>|<0xA6>|理

12 tokens

Byte sequence

Swallow's vocabulary expansion <u>improved Japanese text generation efficiency up to 78%,</u> while almost maintaining accuracy in downstream tasks.

➢ After vocabulary expantion (Swallow-7b-hf)

日本|語|の|自然|言語|処理

6 tokens

Japan|word or language|(*no*-particle)|natural |language|processing

[Fujii+ '24]

Summary of Part 4

- We have looked at:
  - Variations of tokenization strategies and their performance in downstream tasks:
    - For Japanese, not only direct subword tokenization but also a two-step tokenization (using a word segmenter for pre-tokenization) is common, and the latter provides better accuracy for boundary-sensitive tasks such as NER.
    - For Kinyarwanda, a morphology-aware tokenizer designed for this language achieves high accuracy.
  - Common tokenization approaches in current LLMs:
    - Byte-level BPE eliminates the issue of unknown characters.
    - Efficient tokenization design is critical.

Conclusion: Current and Future of Lexical Analysis Tasks

- Decreased importance in the neural era
  - End-to-end neural systems perform well without linguistic information.
  - If the goal is to achieve high accuracy in downstream NLP tasks, considering lexical (and higher-level) linguistic analysis is often unnecessary.
    - However, tokenization-related tasks may enhance downstream task accuracy to some extent in unsegmented and morphologically rich languages.
- Unchanging usefulness as fundamental tools
  - There is stable demand for constructing new corpora with linguistic information for fields such as (computational) linguistics.
  - If the goal is to obtain lexical analysis results, lexical analysis systems are essential.
- Future research for lexical analysis
  - Research opportunities remain since the performance of current systems is not perfect.
  - Academic research in this area will likely continue, though on a small scale.

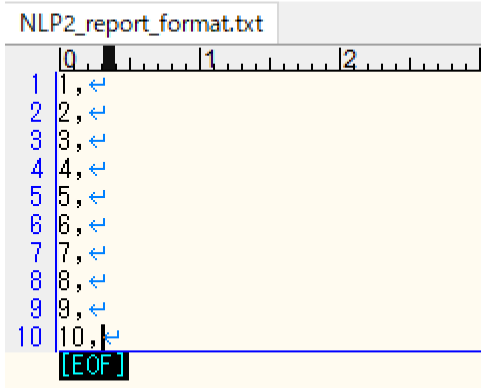
* For conducting further research on established tasks, it is important to explain how critical the errors of existing systems are for a target use case.

Assignments

Updated after the lecture

Report Submission

- Specific Instructions for this session
  - The deadline is 23:59 on **December 12, 2025**.
  - Submit your report in a **plain text file** following the format of NLP2_report_format.txt.



- General Instructions
  - Submit your reports through https://edu-portal.naist.jp/.
  - Late submission policy **after the deadline**:
    - We will accept late submission within 3 weeks, but please use the submission portal with Late Submission.
    - Scores will be scaled: 50% when submitted within 1 week, 25% within 2 weeks and 12.5% within 3 weeks.

Assignments

- Instructions
  - See "NLP2_assignments_20251112.ipynb" and provide answers to questions 1-10.
    I recommend using Google Colab for questions 1-8.

    Find the file in the shared folder

  - For questions that require counts, provide only the numerical value.
    - Example: `5`

  - For questions that choose one options, provide only the option number.
    - Example: `a`

  - Submit your final answers as a single line of text, with answers for Questions 1–10 in order.
    Format each answer as "{question number},{answer}". Spaces between "," are optional.
    - Example:

```
1,5
2,a
```

OR

```
1, 5
2, a
```

<u>Make sure to **follow this format** to ensure your answers are processed correctly!</u>

Appendix

Optional Reading Materials

- Tokenization
  - [Mielke+ 2021] Between words and characters: A Brief History of Open-Vocabulary Modeling and Tokenization in NLP
- Morphological Analysis
  - [Liu 2021] Computational Morphology with Neural Network Approaches
  - [Baxi+ 2024] Recent advancements in computational morphology : A comprehensive survey
- Part-of-Speech Tagging
  - [He+ 2020] A Survey on Recent Advances in Sequence Labeling from Deep Learning Models
    - Note: This paper includes citations for state-of-the-art POS tagging methods (at the time of publication).
  - [Chiche+ 2022] Part of speech tagging: a systematic review of deep learning and machine learning approaches
    - Note: This article mainly reviews journal articles published between 2017-2021 and rarely includes international conference papers, especially those from ACL-related conferences. The coverage of state-of-the-art methods is limited, but it seems useful for understanding a rough trend in POS tagging research.

Optional Reading Materials

- Japanese Morphological Analysis
  - [Unno 2011] 形態素解析の過去・現在・未来 (In Japanese)
  - [Kaji 2013] 日本語形態素解析とその周辺領域における最近の研究動向 (in Japanese)
  - [Kudo 2018] 形態素解析の理論と実装 (Book; In Japanese)
  - [Higashiyama 2022] (slides) Word Segmentation and Lexical Normalization for Unsegmented Languages
    - Note: Sections 2 and 7 discusses preliminaries and future prospectives for neural word segmentation.
- Chinese Word Segmentation
  - [Fu+ 2020] RethinkCWS: Is Chinese Word Segmentation a Solved Task?
  - [Liu+ 2023] Survey on Chinese Word Segmentation
- Multilingual Projects for Resource Construction
  - UniMorph, https://unimorph.github.io/
  - Universal Dependencies, https://universaldependencies.org/