5. Multilingual Text-Editing

Presenter: Eric

5-1. Tokenization

Tokenization

- Token = the smallest unit of text fed to your model
- Unglamorous but of great **practical importance**!
  - If you notice your tokenization is bad, you may need to re-run both pre-training and fine-tuning
  - Particularly important for text **generation** models and for **internationalization**

Tokenization

- Token = the smallest unit of text fed to your model
- Unglamorous but of great **practical importance**!
    - If you notice your tokenization is bad, you may need to re-run both pre-training and fine-tuning
    - Particularly important for text **generation** models and for **internationalization**

- Different levels of tokenization
    - words
    - subwords
    - morphemes
    - characters/bytes

Tokenization

- Token = the smallest unit of text fed to your model
- Unglamorous but of great **practical importance**!
  - If you notice your tokenization is bad, you may need to re-run both pre-training and fine-tuning
  - Particularly important for text **generation** models and for **internationalization**

- Different levels of tokenization
  - words used in the LaserTagger paper
  - subwords used in the Felix and Edit5 papers
  - morphemes
  - characters/bytes

Tokenization Trade-Offs (Text Editing)

Words

```
Untokenized text. ⇒
[“Untokenized”, “text”, “.”]
```

Characters

```
“Untokenized text.” ⇒
[“U”, “n”, “t”, “o”, “k”, “e”,
“n”, “i”, “z”, “e”, “d”, “ “,
“t”, “e”, “x”, “t”, “.”]
```

Tokenization Trade-Offs (Text Editing)

Words

```
Untokenized text. ⇒
[“Untokenized”, “text”, “.”]
```

- Poorly handles morphology

Characters

```
“Untokenized text.” ⇒
[“U”, “n”, “t”, “o”, “k”, “e”,
“n”, “i”, “z”, “e”, “d”, “ “,
“t”, “e”, “x”, “t”, “.”]
```

- NAR decoding can produce nonsense

Tokenization Trade-Offs

Words

```
Untokenized text. ⇒
[“Untokenized”, “text”, “.”]
```

- UNK tokens
- Large vocabulary
- Big embedding matrix
- Many rare words

Characters

```
“Untokenized text.” ⇒
[“U”, “n”, “t”, “o”, “k”, “e”,
“n”, “i”, “z”, “e”, “d”, “ “,
“t”, “e”, “x”, “t”, “.”]
```

- Long-sequences => lower quality
- Slow training and inference
- Non-meaningful units (especially for non-ASCII alphabets)

Tokenization Trade-Offs

Words

```
Untokenized text. ⇒
[“Untokenized”, “text”, “.”]
```

- UNK tokens
- Large vocabulary
- Big embedding matrix
- Many rare words

Characters

```
“Untokenized text.” ⇒
[“U”, “n”, “t”, “o”, “k”, “e”,
“n”, “i”, “z”, “e”, “d”, “ “,
“t”, “e”, “x”, “t”, “.”]
```

- Long-sequences => lower quality
- Slow training and inference
- Non-meaningful units (especially for non-ASCII alphabets)

e.g., ByT5 [Xue et al. 2021], Charformer [Tay et al. 2021]: seq2seq; HCTagger [Gao, Xu, and Shi 2021]

Subword Segmentation

```
Untokenized text. ⇒ [“_Un”, “token”, “ized”, “_text”, “.”]
```

- Different algorithms for optimizing the segmentation:

  BPE, UnigramLM, WordPiece
- Most are reversible: `text == detokenize(tokenize(text))`
  - Original BERT’s WordPiece is not → bad for NLG
- Typical vocabulary size: 30k–250k

Tokenizers Landscape

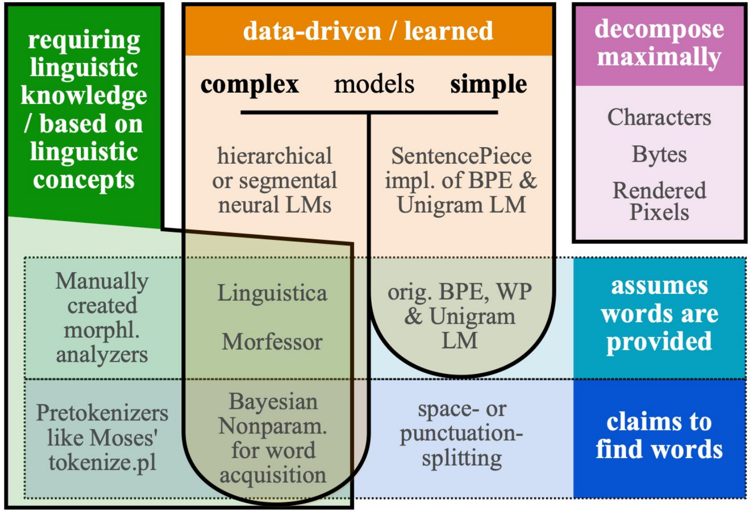


Figure 1: A taxonomy of segmentation and tokenization algorithms and research directions

Mielke et al. *Between words and characters: A Brief History of Open-Vocabulary Modeling and Tokenization in NLP.* arXiv 2021 (pdf)

Different Alphabets and Scripts

Latin (ASCII)

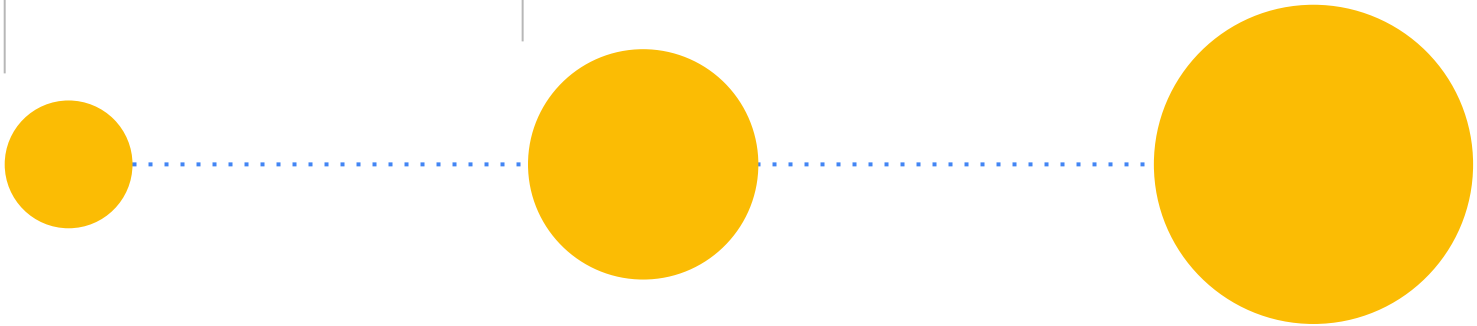
~94 characters

Cyrillic

~200 characters

Korean

~11'000 characters in UTF-8



UNKnown Tokens

1. Subword algorithm is greedy: if "`dog`" is more frequently than "`sei`" or "`생`", we'll prefer it in the vocabulary

UNKnown Tokens

1. Subword algorithm is greedy: if "`dog`" is more frequently than "`sei`" or "`생`", we'll prefer it in the vocabulary
2. Now, how do we encode "`sensei`"?

UNKnown Tokens

1. Subword algorithm is greedy: if "`dog`" is more frequently than "`sei`" or "`생`", we'll prefer it in the vocabulary
2. Now, how do we encode "`sensei`"?
   - We probably have other letters
   - so something like ["`sen`", "`s`", "`e`", "`i`"]

UNKnown Tokens

1. Subword algorithm is greedy: if "`dog`" is more frequently than "`sei`" or "생", we'll prefer it in the vocabulary
2. Now, how do we encode "`sensei`"?
   - We probably have other letters
   - so something like ["`sen`", "`s`", "`e`", "`i`"]
3. But what about "선생"?
   - ["선", "UNK"]

UNKnown Tokens

1. Subword algorithm is greedy: if "dog" is more frequently than "sei" or "생", we'll prefer it in the vocabulary
2. Now, how do we encode "sensei"?
   - We probably have other letters
   - so something like ["sen", "s", "e", "i"]
3. But what about "선생"?
   - ["선", "UNK"]
4. Solution: fallback to bytes
   - ["선", 236, 131, 157]

5-2. Handling Morphology

Editing Morphology

Grammatical Error Correction (GEC) example:

```
Source: "She no drives to market."
```

`Target: "She did ~~no~~ not ~~drives~~ drive to market."`

Depending on tokenization, potentially inefficient `drives->drive` replacement

Morphological Operations

Similar to PRONOMINALIZE tag in sentence fusion, we can introduce a $VERB_FORM_VBZ_VB tag:

- drives -> drive
- goes -> go
- carries -> carry

Omelianchuk et al., 2020 (pdf)

Tag	Example
$KEEP	. . . many people want to travel during the summer . . .
$DELETE	. . . not sure if you are {**you** ⇒ ∅} gifting . . .
$REPLACE_a	. . . the bride wears {**the** ⇒ **a**} white dress . . .
. . .	. . .
$REPLACE_cause	. . . hope it does not {**make** ⇒ **cause**} any trouble . . .
$APPEND_for	. . . he is {**waiting** ⇒ **waiting for**} your reply . . .
. . .	. . .
$APPEND_know	. . . I {**don't** ⇒ **don't know**} which to choose. . .
$CASE_CAPITAL	. . . surveillance is on the {**internet** ⇒ **Internet**} . . .
$CASE_CAPITAL_1	. . . I want to buy an {**iphone** ⇒ **iPhone**} . . .
$CASE_LOWER	. . . advancement in {**Medical** ⇒ **medical**} technology . . .
$CASE_UPPER	. . . the {**it** ⇒ **IT**} department is concerned that. . .
$MERGE_SPACE	. . . insert a special kind of gene {**in to** ⇒ **into**} the cell . . .
$MERGE_HYPHEN	. . . and needs {**in depth** ⇒ **in-depth**} search . . .
$SPLIT_HYPHEN	. . . support us for a {**long-run** ⇒ **long run**} . . .
$NOUN_NUMBER_SINGULAR	. . . a place to live for their {**citizen** ⇒ **citizens**}
$NOUN_NUMBER_PLURAL	. . . carrier of this {**diseases** ⇒ **disease**} . . .
$VERB_FORM_VB_VBZ	. . . going through this {**make** ⇒ **makes**} me feel . . .
$VERB_FORM_VB_VBN	. . . to discuss what {**happen** ⇒ **happened**} in fall . . .
$VERB_FORM_VB_VBD	. . . she sighed and {**draw** ⇒ **drew**} her . . .
$VERB_FORM_VB_VBG	. . . shown success in {**prevent** ⇒ **preventing**} such . . .
$VERB_FORM_VB_VBZ	. . . a small percentage of people {**goes** ⇒ **go**} by bike . . .
$VERB_FORM_VBZ_VBN	. . . development has {**pushes** ⇒ **pushed**} countries to . . .
$VERB_FORM_VBZ_VBD	. . . he {**drinks** ⇒ **drank**} a lot of beer last night . . .
$VERB_FORM_VBZ_VBG	. . . couldn't stop {**thinks** ⇒ **thinking**} about it . . .
$VERB_FORM_VBN_VB	. . . going to {**depended** ⇒ **depend**} on who is hiring . . .
$VERB_FORM_VBN_VBZ	. . . yet he goes and {**eaten** ⇒ **eats**} more melons . . .
$VERB_FORM_VBN_VBD	. . . he {**driven** ⇒ **drove**} to the bus stop and . . .
$VERB_FORM_VBN_VBG	. . . don't want you fainting and {**broken** ⇒ **breaking**} . . .
$VERB_FORM_VBD_VB	. . . each of these items will {**fell** ⇒ **fall**} in price . . .
$VERB_FORM_VBD_VBZ	. . . the lake {**froze** ⇒ **freezes**} every year . . .
$VERB_FORM_VBD_VBN	. . . he has been went {**went** ⇒ **gone**} since last week . . .
$VERB_FORM_VBD_VBG	. . . talked her into {**gave** ⇒ **giving**} me the whole day . . .
$VERB_FORM_VBG_VB	. . . free time, I just {**enjoying** ⇒ **enjoy**} being outdoors . . .
$VERB_FORM_VBG_VBZ	. . . there still {**existing** ⇒ **exists**} many inevitable factors . . .
$VERB_FORM_VBG_VBN	. . . people are afraid of being {**tracking** ⇒ **tracked**} . . .
$VERB_FORM_VBG_VBD	. . . there was no {**mistook** ⇒ **mistaking**} his sincerity . . .

Learned Edit Operations

Idea: instead of learning a vocabulary of **word** replacement, learn vocabulary of **character** replacements

Learned Edit Operations

Idea: instead of learning a vocabulary of **word** replacement, learn vocabulary of **character** replacements

```
,
and
however␣,
,␣but
he
because
,␣although
but
,␣and
although
his
,␣while
it
,␣which
she
```

KEEP

APPEND a
APPEND b
....
APPEND z

REPL. $1^{st} \rightarrow \varnothing$
REPL. $1^{st} \rightarrow$ a
REPL. $1^{st} \rightarrow$ b
....
REPL. $2^{nd} \rightarrow \varnothing$
REPL. $2^{nd} \rightarrow$ a
....
REPL. $5^{th} \rightarrow \varnothing$

UPPERCASE 1^{st}
....

Learned Edit Operations

Idea: instead of learning a vocabulary of **word** replacement, learn vocabulary of **character** replacements

```
,
and
however␣,
,␣but
he
because
,␣although
but
,␣and
although
his
,␣while
it
,␣which
she
```

```
KEEP

APPEND a
APPEND b
........
APPEND z

REPL. 1st → ∅
REPL. 1st → a
REPL. 1st → b
....
REPL. 2nd → ∅
REPL. 2nd → a
....
REPL. 5th → ∅

UPPERCASE 1st
....
```

Source: gatherin leafes

["_gathe", "rin", "_lea", "fes"]

Target: Gathering leaves

Tags: [UPP.2^{nd}, APP.g, KEEP, REPL.1^{st} → v]

Realize: ["_Gathe", "ring", "_lea", "ves"]

Straka et al., 2021 (pdf)

5-3. Practical Aspects of Multilingual Models

Per-Language Model (vs. Multilingual)

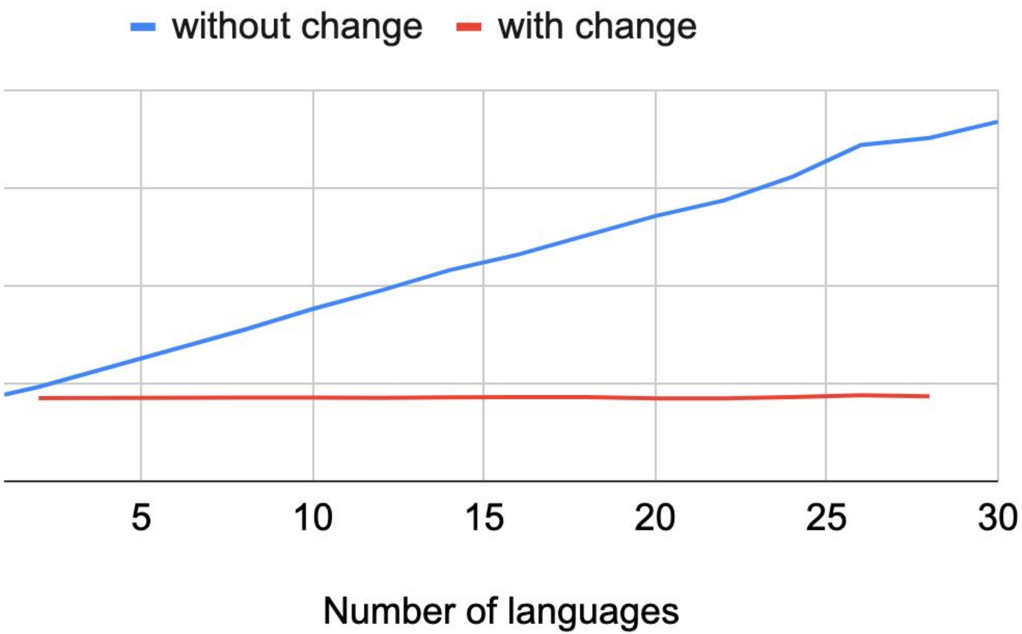
Per-language

+ better alphabet => relying less on byte fallback (e.g., KR-BERT, RuBERT)
+ smaller model
+ independent release cycle

Multilingual

+ cross-lingual learning
+ simpler training
+ lower maintenance costs
    + lower complexity
    + lower resource (TPU/RAM) footprint

Per-Language Edit Operations



A change to introduce a separate softmax layer for LaserTagger per language. TPU Inference time (scale)

Encoder Vocabulary & Tokenization

One size does not fit all:

- Bigger [SentencePiece] vocabulary => smaller sequence length => faster encoding
- ... but it can make the source/target alignment harder
- ... and it makes the model bigger
- ... and languages need to be properly balanced

See Chung et al. (2020) [pdf] on how to merge vocabularies

Rachel Claire on pexels.com (link)

Questions?