Sequence Tagging with Contextual and Non-Contextual Subword Representations
A Multilingual Evaluation

Benjamin Heinzerling and Michael Strube

ACL 2019
Multilingual Subword Representations

fastText (Bojanovski+’17) 294 languages

BPEmb (Heinzerling+’18) 275 languages

BERT 🐦 (Devlin+’19) 104 languages
Multilingual Subword Representations

*fastText* (Bojanovski+’17)  294 languages*

BPEmb (Heinzerling+’18)  275 languages*

BERT 🦙 (Devlin+’19)  104 languages*

WHAT DO I CHOOSE?

TOO MANY OPTIONS!
Multilingual Subword Representations

*fastText* (Bojanovski+’17) 294 languages*

BPEmb (Heinzerling+’18) 275 languages*

BERT 🧡 (Devlin+’19) 104 languages*

Goal of this work: Help you make this decision for NER and POS tagging
**Multilingual Subword Representations**

<table>
<thead>
<tr>
<th>Model</th>
<th>Languages</th>
</tr>
</thead>
<tbody>
<tr>
<td>fastText (Bojanovski+’17)</td>
<td>294 languages*</td>
</tr>
<tr>
<td>BPEmb (Heinzerling+’18)</td>
<td>275 languages*</td>
</tr>
<tr>
<td>BERT (Devlin+’19)</td>
<td>104 languages*</td>
</tr>
</tbody>
</table>

Goal of this work: Help you make this decision for NER and POS tagging

FastText (Bojanovski+’17)
Word embedding = sum of char-ngram embeddings

Magnus Carlsen played Viswanathan Anand

$$\text{viswanathan} = \text{vis} + \text{isw} + \text{swa} + \text{wan} + \text{ana} + \text{nat} + \text{ath} + \text{tha} + \text{han} +$$
$$\text{visw} + \text{iswa} + \text{swan} + \text{wana} + \text{anat} + \text{nath} + \text{atha} + \text{than} + \text{viswa} +$$
$$\text{iswan} + \text{swana} + \text{wanat} + \text{anath} + \text{natha} + \text{athan} + \text{viswan} + \text{iswana} +$$
$$\text{swanat} + \text{wanath} + \text{anatha} + \text{nathan}$$

Many char-ngrams → huge file sizes

Check out tiny FastText: "Subword-based Compact Reconstruction of Word Embeddings" (Sasaki, Suzuki, Inui, NAACL’19)
Byte-Pair Encoding (Sennrich+’16)
Iteratively merge the most frequent pair of adjacent symbols

\[ \text{the netherlands are neither here nor there} \]
Byte-Pair Encoding (Sennrich+’16)
Iteratively merge the most frequent pair of adjacent symbols

\textit{the netherlands are neither here nor there}

1. \texttt{the\_netherlands\_are\_neither\_here\_nor\_there}
Byte-Pair Encoding (Sennrich+’16)
Iteratively merge the most frequent pair of adjacent symbols

the netherlands are neither here nor there

1. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e
Byte-Pair Encoding (Sennrich+’16)

Iteratively merge the most frequent pair of adjacent symbols

*the netherlands are neither here nor there*

1. the_netherlands_are_neither_here_nor_there
2. the_netherlands_are_neither_here_nor_there
3. the_netherlands_are_neither_here_nor_there
4. the_netherlands_are_neither_here_nor_there
5. the_netherlands_are_neither_here_nor_there
6. the_netherlands_are_neither_here_nor_there

BPE vocab: he, the, ther, ne, re

With 5000 merge operations learned on Wikipedia:

*the netherlands are neither here nor there*
Byte-Pair Encoding (Sennrich+’16)
Iteratively merge the most frequent pair of adjacent symbols

*the netherlands are neither here nor there*

1. the_netherlands_are_neither_here_nor_there
2. the_netherlands_are_neither_here_nor_there
3. the_netherlands_are_neither_here_nor_there
4. the_netherlands_are_neither_here_nor_there
5. the_netherlands_are_neither_here_nor_there
6. the_netherlands_are_neither_here_nor_there

BPE vocab: he, the, ther, ne, re

With 5000 merge operations learned on Wikipedia:

*the netherlands are neither here nor there*
Byte-Pair Encoding (Sennrich+'16)
Iteratively merge the most frequent pair of adjacent symbols

the netherlands are neither here nor there

1. the_netherlands_are_neither_here_nor_there
2. the_netherlands_are_neither_here_nor_there
3. the_netherlands_are_neither_here_nor_there

BPE vocab: he, the, ther, ne, re

With 5000 merge operations learned on Wikipedia:

the_netherlands_are_neither_here_nor_there
Byte-Pair Encoding (Sennrich+’16)
Iteratively merge the most frequent pair of adjacent symbols

the netherlands are neither here nor there

1. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e
2. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e
3. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e
4. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e
5. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e

BPE vocab: he, the, ther, ne, re

With 5000 merge operations learned on Wikipedia:
the _ nether lands _ are _ ne ither _ here _ nor _ there
Byte-Pair Encoding (Sennrich+’16)

Iteratively merge the most frequent pair of adjacent symbols

the netherlands are neither here nor there

1. the_nether_lands_are_neither_here_nor_there
2. the_nether_lands_are_neither_here_nor_the_re
3. the_nether_lands_are_neither_here_nor_the_re
4. the_nether_lands_are_neither_here_nor_the_re
Byte-Pair Encoding (Sennrich+’16)
Iteratively merge the most frequent pair of adjacent symbols

the netherlands are neither here nor there

1. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e re
2. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e re
3. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e re
4. the _ _ n e _ t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e re

BPE vocab: he, the, ther, ne, re
With 5000 merge operations learned on Wikipedia:
the _ nether lands _ are _ ne ither _ here _ nor _ there
Byte-Pair Encoding (Sennrich+’16)
Iteratively merge the most frequent pair of adjacent symbols

the netherlands are neither here nor there

1. the_netherlands_are_neither_here_nor_there
2. the_netherlands_are_neither_here_nor_the_re
3. the_netherlands_are_neither_here_nor_the_re
4. the_netherlands_are_neither_here_nor_the_re
5. the_netherlands_are_neither_here_nor_the_re

BPE vocab: he, the, ther, ne, re

With 5000 merge operations learned on Wikipedia:

the_netherlands_are_neither_here_nor_there
Byte-Pair Encoding (Sennrich+’16)
Iteratively merge the most frequent pair of adjacent symbols

*the netherlands are neither here nor there*

1. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e
2. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e
3. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e
4. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e
5. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e
Byte-Pair Encoding (Sennrich+’16)
Iteratively merge the most frequent pair of adjacent symbols

1. the_netherlands_are_neither_here_nor_there
2. the_netherlands_are_neither_here_nor_there
3. the_netherlands_are_neither_here_nor_there
4. the_netherlands_are_neither_here_nor_there
5. the_netherlands_are_neither_here_nor_there
6. the_netherlands_are_neither_here_nor_there

the netherlands are neither here nor there
Byte-Pair Encoding (Sennrich+’16)

Iteratively merge the most frequent pair of adjacent symbols

the netherlands are neither here nor there

1. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e
2. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e
3. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e
4. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e
5. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e
6. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e

BPE vocab: he, the, ther, ne, re
Byte-Pair Encoding (Sennrich+’16)

Iteratively merge the most frequent pair of adjacent symbols

the netherlands are neither here nor there

1. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e
2. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e
3. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e
4. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e
5. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e
6. the _ n e t h e r l a n d s _ a r e _ n e i t h e r _ h e r e _ n o r _ t h e r e

BPE vocab: he, the, ther, ne, re

With 5000 merge operations learned on Wikipedia:
the _ nether lands _ are _ ne ither _ here _ nor _ there
BPEmb (Heinzerling+’18, shameless plug)

BPE + GloVe = Byte-Pair Embeddings (http://nlp.h-its.org/bpemb)

Advantages: Easy to use, small file sizes, no tokenization required
Disadvantage: not contextual (trained with GloVe, no LM objective)

Import BPEmb:

```python
>>> from bpemb import BPEmb
```

Load a BPEmb model for English:

```python
>>> bpemb_en = BPEmb(lang="en")
```

Byte-pair encode text:

```python
>>> bpemb_en.encode("Stratford")
['_strat', 'ford']
>>> bpemb_en.encode("This is anarchism")
['_this', '_is', '_an', 'arch', 'ism']
```

Load a Chinese model with vocabulary size 100,000:

```python
>>> bpemb_zh = BPEmb(lang="zh", vs=100000)
>>> bpemb_zh.encode("这是一个中文句子") # "This is a Chinese sentence."
['_这', '_是', '中文', '句子'] # ["This is a", "Chinese", "sentence"]
```
Multilingual BERT (Devlin+’19)
Contextual subword embeddings with shared 104-lingual vocabulary

BPE is language-agnostic*: can give it any character sequence

Multilingual Bert recipe:

1. Train on multilingual texts → get shared multilingual subword vocabulary (100k)
2. Subword-encode texts with this shared vocabulary
3. Train BERT on encoded texts

* But not language-independent (see discussion in Appendix 1)
### Overview: Subword segmentation methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Subword segmentation and token transformation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original text</td>
<td>Magnus Carlsen played against Viswanathan Anand</td>
</tr>
</tbody>
</table>
## Overview: Subword segmentation methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Subword segmentation and token transformation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original text</td>
<td>Magnus Carlsen played against Viswanathan Anand</td>
</tr>
<tr>
<td>FastText</td>
<td>magnus+mag+... carlsen+car+arl+... played+... against+... vis+isw+...+nathan ana+...</td>
</tr>
</tbody>
</table>
## Overview: Subword segmentation methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Subword segmentation and token transformation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original text</td>
<td>Magnus Carlsen played against Viswanathan Anand</td>
</tr>
<tr>
<td>FastText</td>
<td>magnus+mag+. . . carlsen+car+arl+. . . played+. . . against+. . . vis+isw+. . . +nathan ana+. . .</td>
</tr>
<tr>
<td>BPE vs1000</td>
<td>_m ag n us _car l s en _play ed _against _v is w an ath an _an and</td>
</tr>
<tr>
<td>BPE vs10000</td>
<td>_magn us _car ls en _played _against _vis wan ath an _an and</td>
</tr>
<tr>
<td>BPE vs100000</td>
<td>_magnus _carlsen _played _against _viswanathan _anand</td>
</tr>
</tbody>
</table>
## Overview: Subword segmentation methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Subword segmentation and token transformation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original text</td>
<td>Magnus Carlsen played against Viswanathan Anand</td>
</tr>
<tr>
<td>FastText</td>
<td>magnus+mag+. . . carlsen+car+arl+. . . played+. . . against+. . . vis+isw+. . . +nathan ana+. . .</td>
</tr>
<tr>
<td>BPE vs1000</td>
<td>_m ag n us _car ls en _play ed _against _v is w an ath an _an and</td>
</tr>
<tr>
<td>BPE vs10000</td>
<td>_magn us _car ls en _played _against _vis wan athan _an and</td>
</tr>
</tbody>
</table>
Overview: Subword segmentation methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Subword segmentation and token transformation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original text</td>
<td>Magnus Carlsen played against Viswanathan Anand</td>
</tr>
<tr>
<td>FastText</td>
<td>magnus+mag+... carlsen+car+arl+... played+... against+... vis+isw+...+nathan ana+...</td>
</tr>
<tr>
<td>BPE vs1000</td>
<td>_m ag n us _car l s en _play ed _against _v is w an ath an _an and</td>
</tr>
<tr>
<td>BPE vs10000</td>
<td>_magn us _car ls en _played _against _vis wan athan _an and</td>
</tr>
<tr>
<td>BPE vs100000</td>
<td>_magnus _carlsen _played _against _viswan athan _anand</td>
</tr>
</tbody>
</table>
## Overview: Subword segmentation methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Subword segmentation and token transformation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original text</td>
<td>Magnus Carlsen played against Viswanathan Anand</td>
</tr>
<tr>
<td>FastText</td>
<td>magnus+mag+. . . carlsen+car+ar+. . . played+. . . against+. . . vis+is+. . . +nathan ana+. . .</td>
</tr>
<tr>
<td>BPE vs1000</td>
<td>_m ag n us _car l s en _play ed _against _vis wan athan _an and</td>
</tr>
<tr>
<td>BPE vs10000</td>
<td>_magn us _car ls en _played _against _vis wan athan _an and</td>
</tr>
<tr>
<td>BPE vs100000</td>
<td>_magnus _carlsen _played _against _viswan athan _anand</td>
</tr>
<tr>
<td>BERT*</td>
<td>Magnus Carl ##sen played against V ##is ##wana ##than Anand</td>
</tr>
</tbody>
</table>
## Overview: Subword segmentation methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Subword segmentation and token transformation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original text</td>
<td>Magnus Carlsen played against Viswanathan Anand</td>
</tr>
<tr>
<td>FastText</td>
<td>magnus+mag+... carlsen+car+arl+... played+... against+... vis+isw+...+nathan+... ana+...</td>
</tr>
<tr>
<td>BPE vs1000</td>
<td>_m ag n us _car l s en _play ed _against _v is w an ath an _an and</td>
</tr>
<tr>
<td>BPE vs10000</td>
<td>_magn us _car ls en _played _against _vis wan athan _an and</td>
</tr>
<tr>
<td>BPE vs100000</td>
<td>_magnus _carlsen _played _against _viswan athan _anand</td>
</tr>
<tr>
<td>BERT*</td>
<td>Magnus Carl ##sen played against V ##is ##wana ##than Anand</td>
</tr>
<tr>
<td>Word shape</td>
<td>Aa Aa a a Aa Aa</td>
</tr>
</tbody>
</table>

*BERT* denotes the use of BERT in subword segmentation.
Dataset: WikiAnn (Pan+’17)
NER annotations in 282 languages

Così, dopo una tappa a Piacenza, si diresse verso Firenze.
[So, after a stop in Piacenza, he headed for Florence.]*

Bu kişilere örnek olarak devrin ünlü Floransa’lı şairi Guido Cavalcanti’yi verebiliriz.
[For example, the famous Florentine poet of the time, Guido Cavalcanti.]*

*Translation according to Google translate

Best (only!?) system on WikiAnn by the authors (“Pan17”):

- Crosslingual gazetters (Firenze = Floransa)
- Morphological features (Floransa’lı → Floransa, so “lı” is a suffix)
- LSTM sequence tagger
NER Experiments: FastText vs. BPEmb vs. BERT
Which subword representation is best for multilingual NER?

Setup: Train one model for
- each subword representation
- and each language
- with crossvalidation
NER Experiments: FastText vs. BPEmb vs. BERT

Which subword representation is best for multilingual NER?

Setup: Train one model for

- each subword representation
- and each language
- with crossvalidation

Model architecture:

- LSTM sequence tagger, no frills
NER Experiments: FastText vs. BPEmb vs. BERT

Which subword representation is best for multilingual NER?

Setup: Train one model for

- each subword representation
- and each language
- with crossvalidation

Model architecture:

- LSTM sequence tagger, no frills
- no crosslingual or morphological features
NER Experiments: FastText vs. BPEmb vs. BERT

Which subword representation is best for multilingual NER?

Setup: Train one model for

- each subword representation
- and each language
- with crossvalidation

Model architecture:

- LSTM sequence tagger, no frills
- no crosslingual or morphological features
- no CRF (Too slow. Hogging the GPUs even longer would have gotten me killed by the angry mob in the cluster queue)
NER Experiments: FastText vs. BPEmb vs. BERT

Which subword representation is best for multilingual NER?

Setup: Train one model for

- each subword representation
- and each language
- with crossvalidation

Model architecture:

- LSTM sequence tagger, no frills
- no crosslingual or morphological features
- no CRF (Too slow. Hogging the GPUs even longer would have gotten me killed by the angry mob in the cluster queue)
NER Experiments: FastText vs. BPEmb vs. BERT

Which subword representation is best for multilingual NER?

Setup: Train one model for

- each subword representation
- and each language
- with crossvalidation

Model architecture:

- LSTM sequence tagger, no frills
- no crosslingual or morphological features
- no CRF (Too slow. Hogging the GPUs even longer would have gotten me killed by the angry mob in the cluster queue)

Only NER results in this talk, also did POS tagging, similar trends
NER on 265 languages: Pan17 is best
FastText worst, BPEmb almost as good
Characters are still useful with BPE
Better than word shape in high-res languages

Average F1
Pan17
BPEmb
BPEmb+shape
BPEmb+char

Languages
70
75
80
85
90
95
100

Multilingual BERT works well

Better with characters, best with BPEmb+characters
How to do better on low-res languages?

Hypothesis: Cross-lingual transfer should help

So let's train multilingual subword embeddings!
How to do better on low-res languages?

Hypothesis: Cross-lingual transfer should help
How to do better on low-res languages?

Hypothesis: Cross-lingual transfer should help

So let’s train multilingual subword embeddings!
MultiBPEmb
Non-contextual $\rightarrow$ large vocab size, \url{https://nlp.h-its.org/bpemb/multi/}

- Language Modeling objective in BERT and other muppets limits vocab size
MultiBPEmb

Non-contextual $\rightarrow$ large vocab size, \url{https://nlp.h-its.org/bpemb/multi/}

- Language Modeling objective in BERT and other muppets limits vocab size
- LM involves softmax over vocab, becomes slower as vocab grows
MultiBPEmb

Non-contextual $\rightarrow$ large vocab size, https://nlp.h-its.org/bpemb/multi/

- Language Modeling objective in BERT and other muppets limits vocab size
- LM involves softmax over vocab, becomes slower as vocab grows
- No LM in non-contextual embeddings $\rightarrow$ go crazy with vocab size
MultiBPEmb

Non-contextual $\rightarrow$ large vocab size, https://nlp.h-its.org/bpemb/multi/

- Language Modeling objective in BERT and other muppets limits vocab size
- LM involves softmax over vocab, becomes slower as vocab grows
- No LM in non-contextual embeddings $\rightarrow$ go crazy with vocab size
- BPE vocab sizes: 100k, 320k, 1000k
MultiBPEmb
Non-contextual $\rightarrow$ large vocab size, https://nlp.h-its.org/bpemb/multi/

- Language Modeling objective in BERT and other muppets limits vocab size
- LM involves softmax over vocab, becomes slower as vocab grows
- No LM in non-contextual embeddings $\rightarrow$ go crazy with vocab size
- BPE vocab sizes: 100k, 320k, 1000k
- Train one embedding model on all languages in Wikipedia
MultiBPEmb

Non-contextual → large vocab size, https://nlp.h-its.org/bpemb/multi/

- Language Modeling objective in BERT and other muppets limits vocab size
- LM involves softmax over vocab, becomes slower as vocab grows
- No LM in non-contextual embeddings → go crazy with vocab size
- BPE vocab sizes: 100k, 320k, 1000k
- Train one embedding model on all languages in Wikipedia
MultiBPEmb

Non-contextual $\rightarrow$ large vocab size, https://nlp.h-its.org/bpemb/multi/

- Language Modeling objective in BERT and other muppets limits vocab size
- LM involves softmax over vocab, becomes slower as vocab grows
- No LM in non-contextual embeddings $\rightarrow$ go crazy with vocab size
- BPE vocab sizes: 100k, 320k, 1000k
- Train one embedding model on all languages in Wikipedia

You can use MultiBPEmb in Python like this:

```python
>>> from bpemb import BPEmb
>>> multibpemb = BPEmb(lang="multi", vs=1000000, dim=300)
>>> text = 'John F. Kennedy said "Ich bin ein Pfannkuchen". 这是一个中文句子。日本語の文章です。'
>>> subwords = multibpemb.encode(text)
>>> print(" ".join(subwords))
_john_f . _kennedy _said _" ich _bin _ein _pfann kuchen " . _这 是一个 中文 句子 。 _日本 語の 文章 です 。
```
Bigger multilingual subword vocab is better
Two percent higher average NER F1 score on dev of all languages
Multilingual training allows crosslingual transfer
Train one model on concatenation of NER training data in all languages

finetune: multilingual pretraining, then train on one language only
265-lingual semantic space? No.
MultiBPEmb embedding space before (l) and after (r) NER training
265-lingual semantic space? No.

MultiBPEmb embedding space before (l) and after (r) NER training

Color = unicode codepoint ($\approx$ language families)
265-lingual semantic space? No.

MultiBPEmb embedding space before (l) and after (r) NER training

![Scatter plots showing embedding spaces before and after NER training.](image)

- Color = unicode codepoint (≈ language families)
- Does not suggest crosslingual transfer

---

Check out: Massively Multilingual Transfer for NER (Rahimi, Li, and Cohn, ACL '19)
265-lingual semantic space? No.

MultiBPEmb embedding space before (l) and after (r) NER training

- Color = unicode codepoint ($\approx$ language families)
  - Does not suggest crosslingual transfer
  - Check out: Massively Multilingual Transfer for NER (Rahimi, Li, and Cohn, ACL’19)
265-lingual semantic space? No.

MultiBPEmb embedding space before (l) and after (r) NER training

Color = unicode codepoint (≈ language families)

- Does not suggest crosslingual transfer
- Check out: Massively Multilingual Transfer for NER (Rahimi, Li, and Cohn, ACL’19)
- Improvements more likely due to the multilingual setting: enables the model to better learn BIO constraints, tag distributions
Conclusions: So which subword embeddings are best?

- It depends...

- Combining representations = Best ("a bit disappointing" according to a reviewer)

- Multilingual BERT = surprisingly good

- BPEmb isn’t bad either, also less resource-hungry

- Character embeddings still useful, both with BPEmb and BERT

- Multilingual BERT’s small vocab size probably suboptimal

- Multilingual pretraining monolingual finetuning = Awesome for low-res

Thank You!
Conclusions: So which subword embeddings are best?

- It depends...
- Combining representations = Best ("a bit disappointing" according to a reviewer)
Conclusions: So which subword embeddings are best?

▶ It depends...
▶ Combining representations = Best ("a bit disappointing" according to a reviewer)
▶ Multilingual BERT = surprisingly good
Conclusions: So which subword embeddings are best?

- It depends...
- Combining representations = Best ("a bit disappointing" according to a reviewer)
- Multilingual BERT = surprisingly good
- BPEmb isn’t bad either, also less resource-hungry
Conclusions: So which subword embeddings are best?

- It depends...
- Combining representations = Best ("a bit disappointing" according to a reviewer)
- Multilingual BERT = surprisingly good
- BPEmb isn’t bad either, also less resource-hungry
- Character embeddings still useful, both with BPEmb and BERT
Conclusions: So which subword embeddings are best?

- It depends...
- Combining representations = Best ("a bit disappointing" according to a reviewer)
- Multilingual BERT = surprisingly good
- BPEmb isn’t bad either, also less resource-hungry
- Character embeddings still useful, both with BPEmb and BERT
- Multilingual BERT’s small vocab size probably suboptimal
Conclusions: So which subword embeddings are best?

- It depends...
- Combining representations = Best ("a bit disappointing" according to a reviewer)
- Multilingual BERT = surprisingly good
- BPEmb isn’t bad either, also less resource-hungry
- Character embeddings still useful, both with BPEmb and BERT
- Multilingual BERT’s small vocab size probably suboptimal
- Multilingual pretraining monolingual finetuning = Awesome for low-res
Conclusions: So which subword embeddings are best?

- It depends...
- Combining representations = Best ("a bit disappointing" according to a reviewer)
- Multilingual BERT = surprisingly good
- BPEmb isn’t bad either, also less resource-hungry
- Character embeddings still useful, both with BPEmb and BERT
- Multilingual BERT’s small vocab size probably suboptimal
- Multilingual pretraining monolingual finetuning = Awesome for low-res
Conclusions: So which subword embeddings are best?

- It depends...
- Combining representations = Best ("a bit disappointing" according to a reviewer)
- Multilingual BERT = surprisingly good
- BPEmb isn’t bad either, also less resource-hungry
- Character embeddings still useful, both with BPEmb and BERT
- Multilingual BERT’s small vocab size probably suboptimal
- Multilingual pretraining monolingual finetuning = Awesome for low-res

Thank You!