LLM Sys

11868 LLM Systems
Tokenization

Lei Li

Carnegie Mellon University
Language Technologies Institute

Recap – Key Ideas in Modern Pre-trained LLMs

- Pretraining: Mask-labeled recovery of random spans + next token prediction
- Multitask supervised fine-tuning with instruction templates
- Relative position instead of absolute position: Rotary positional embedding
- Smooth activation: SwishGLU
- Sparse attention patterns

Outline

- Subword tokenization: Byte-Pair-Encoding
  - Code walk through
- Information-theoretic vocabulary (VOLT)
- Practical Considerations in LLM
- Vocabulary sharing and impact on multilingual performance
- Tokenizer-free model (BLT)

Tokenizer – split text into basic units

Many words don’t map to one token: indivisible.

↓ tokenizer

`Many words don't map to one token: indivisible.`

`[7085, 2456, 836, 470, 3975, 284, 530, 11241, 25, 773, 452, 12843, 13]`

↓ embedding table lookup

2.3	-3.2	8.3	5.4	2.1	3.9	-8.9	3.8	3.9	3.3
4.5	5.9	4.5	7.1	1.0	5.3	5.0	3.1	0.7	5.0
…	…	…	…	…	…	…	…	…	…
3.8	1.2	3.8	9.0	9.3	3.1	4.2	0.8	9.2	5.8

Simple Tokenization – Word-level

- Word-level Tokenization
    - Break by space and punctuation.
    - English, French, German, Spanish

The most eager is Oregon which is enlisting 5,000 drivers in the country's biggest experiment.

- - Special treatment: numbers replaced by special token [number]
    - How large is the Vocabulary? Cut-off by frequency, the rest replaced by [UNK]

Vocabulary – simple example

```
from collections import defaultdict

def build_word_dict(file_path):
    word_dict = defaultdict(int)
    with open(file_path, 'r', encoding='utf-8') as file:
        for line in file:
            words = line.split()  # Split by spaces
            for word in words:
                word_dict[word] += 1

    return word_dict
```

What is a word?

How many words?

Bob's handyman is a do-it-yourself kinda guy, isn't he?

Clitic

noun-noun compound

multi-word expression

contraction

Words

- Orthographic definition
  - strings separated by white spaces
  - spoken language: units corresponding to written word separated by pause
  - problem: Bob's handy man is a do-it-yourself kinda guy, isn't he?
- What about languages that do not use white spaces?

他昨天晚上去看了消失的她

he yesterday night watched lost in stars

Pros and Cons of Word-level Tokenization

- Easy to implement

- Cons:
  - Out-of-vocabulary (OOV) or unknown tokens, e.g. Covid
  - Tradeoff between parameters size and unknown chances.
    - Smaller vocab => fewer parameters to learn, easier to generate (deciding one word from smaller dictionary), more OOV
    - Larger vocab => more parameters to learn, harder to generate, less OOV
  - Hard for certain languages with continuous script: Japanese, Chinese, Korean, Khmer, etc. Need separate word segmentation tool (can be neural networks)

Character-level Tokenization

T	h	e	m	o	s	t	e	a	g	e	r	i	s	O	r	e	g	…

- Each letter and punctuation is a token
- Pros:
  - Very small vocabulary (except for some languages, e.g. Chinese)
  - No Out-of-Vocabulary token
- Cons:
  - A sentence can be longer sequence
  - Tokens do not representing semantic meaning

Subword-level Tokenization

The | most | eager | is | Oregon | which | is | en | listing | 5,000 | driver | s | in | the | country | ’s | big | g | est | experiment | .

- Goal:
  - moderate size vocabulary
  - no OOV

- Idea:
  - represent rare words (OOV) by sequence of subwords

- Byte Pair Encoding (BPE)
  - not necessarily semantic meaningful
  - Originally for data compression Philip Gage. A New Algorithm for Data Compression, 1994

Byte Pair Encoding (BPE) for Text Building Vocabulary

1. Initialize vocabulary with all characters as tokens (also add end-of-word symbol) and frequencies

2. Loop until vocabulary size reaches capacity
   1) Count successive pairs of tokens in corpus
   2) Rank and select the top frequent pair
   3) Combine the pair to form a new token, add to vocabulary

3. Output final vocabulary

Rico Sennrich et al. Neural Machine Translation of Rare Words with Subword Units. 2016

Byte-Pair-Encoding Tokenization

1. starting from chars
2. repeatedly, merge most frequent pairs to form new tokens
3. until reaching a fixed size.

raw word	freq.
cat	90
catch	50
rat	80
rattle	40

a
c
e
h
l
t

merge(‘a’, ’t’)

a
c
e
h
l
t
at

merge (‘c’, ’at’)

a
c
e
h
l
t
at
cat

merge (‘r’, ’at’)

a
c
e
h
l
t
at
cat
rat

merge (‘cat’, ’c’)

a
c
e
h
l
t
at
cat
rat
catc

Neural Machine Translation of Rare Words with Subword Units. Sennrich et al. ACL 2016

BPE Tokenization

- Split text by space or other delimiters

- Repeat
  - greedy find the longest prefix that matches a token in BPE dictionary
  - split and process the remaining parts until no more text left

Code Example

- https://github.com/llmsystem/llmsys_code_examples/blob/main/tokenization/tokenization.ipynb

Outline

- Subword tokenization: Byte-Pair-Encoding
  - Code walk through
- Information-theoretic vocabulary (VOLT)
- Practical Considerations in LLM
- Vocabulary sharing and impact on multilingual performance
- Tokenizer-free model (BLT)

Measuring Vocabulary

- Compression
  - average number of bytes per token

$$BPT = \frac{\text{\# utf8 bytes}}{\text{\#tokens}}$$

  - normalized sequence length

$$NSL = \frac{\#tokens}{\#tokens\ in\ LLaMA}$$

  - normalized entropy (next)

Find Optimal Vocabulary

Numerous possible vocabularies at the sub-word level.

Normalized Entropy

😁

Vocab
1k tokens

Vocab
10k tokens

Vocab
30k tokens

Size

😁

Which one leads to better NLG/MT performance?

Repeated full training and testing are required to find the optimal vocabulary!(BPE-Search)

Xu, Zhou, Gan, Zheng, **Li**. Vocabulary Learning via Optimal Transport for Neural Machine Translation. ACL 2021a.

- Normalized Entropy (modified based on Information Entropy) $\mathcal{H}(v) = -\frac{1}{l_v}\sum_{i\in v} P(i)logP(i)$

l_v: average number of chars for v's all tokens

- It measures semantic-information-per-char
  - Smaller is favorable. Less ambiguity and easy to generate

VOLT: Using entropy to learn vocabulary

- Normalized Entropy (modified based on Information Entropy) $\mathcal{H}(v) = -\frac{1}{l_v}\sum_{i\in v} P(i)logP(i)$

token prob.

l_v: average number of chars for v's all tokens

- It me antic-information-per-char
  - Sm able. Less ambiguity and ea ate

Token	count
a	200
e	90
c	30
t	30
s	90

$\mathcal{H}(v) = 1.37$

Token	count
a	100
aes	90
cat	30

$\mathcal{H}(v) = 0.14$

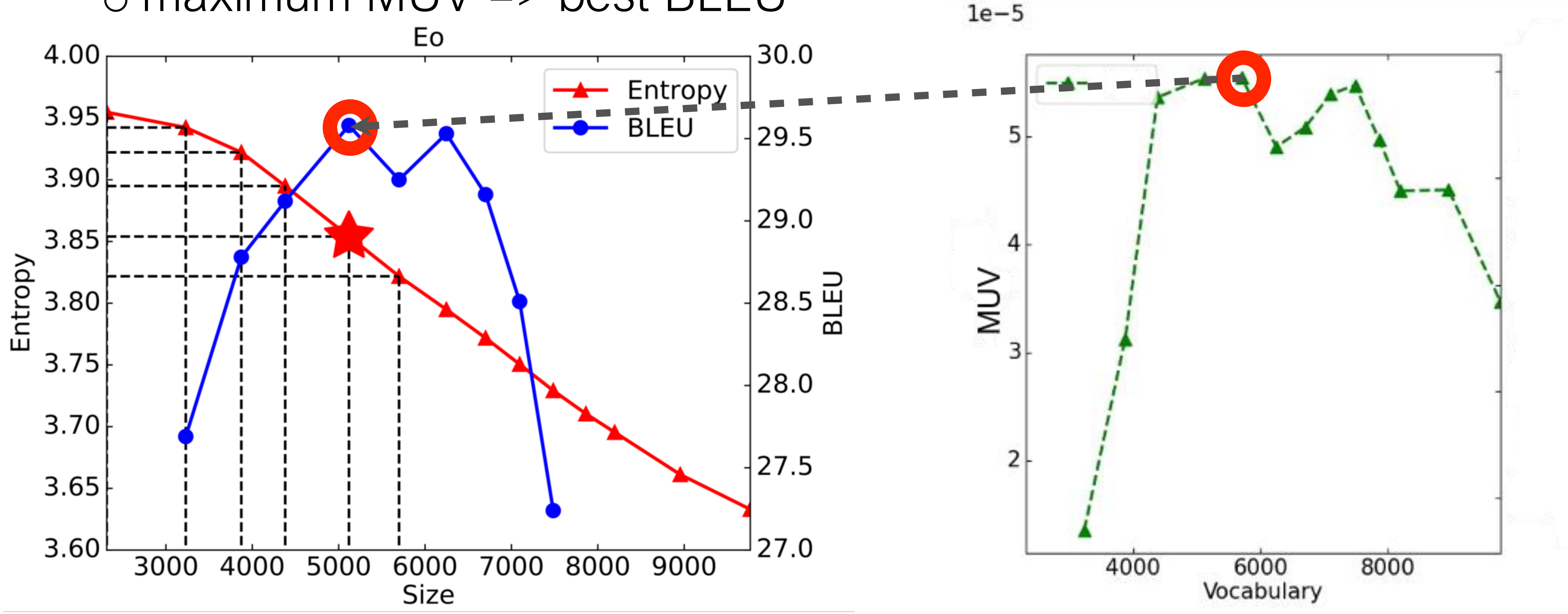
MUV: Utility of Information for Adding Tokens

- Value: Normalized Entropy
- Cost: Size
- Marginal Utility of information for Vocabulary (MUV)
  - $M_{v_k \to v_{k+m}} = -\frac{H(v_k) - H(v_{k+m})}{m}$
  - Negative gradients of normalized entropy to size
  - How much value each token brings

Xu, Zhou, Gan, Zheng, **Lei Li**. Vocabulary Learning via Optimal Transport for Neural Machine Translation. ACL 2021a

MUV is good indicator for MT performance

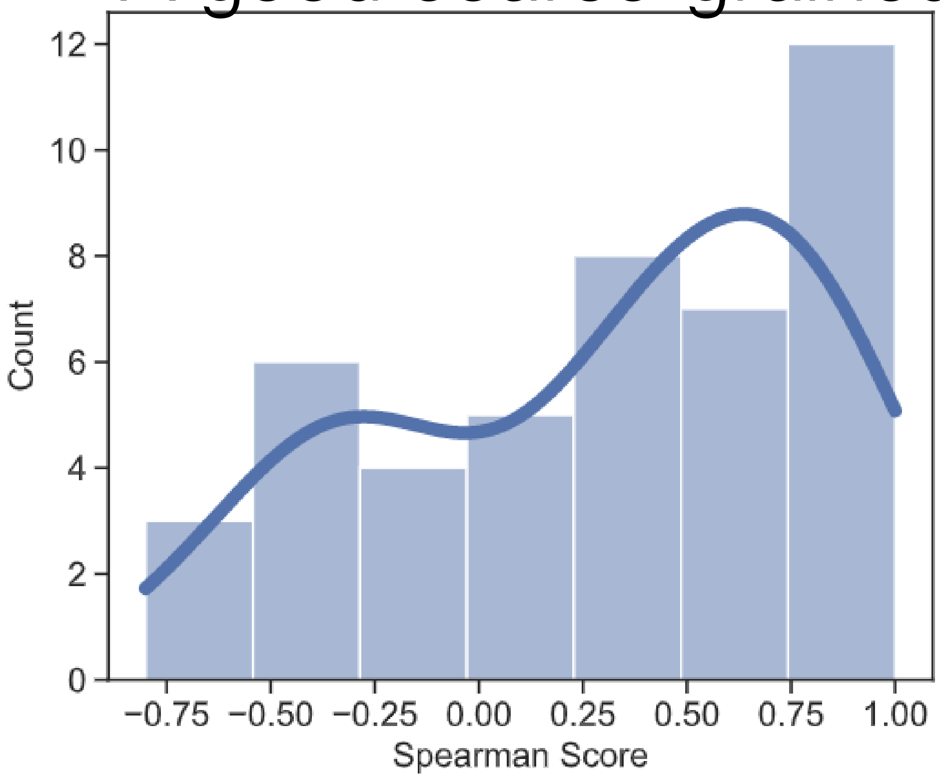
- Cost-effective point in MUV curve
    - maximum MUV => best BLEU

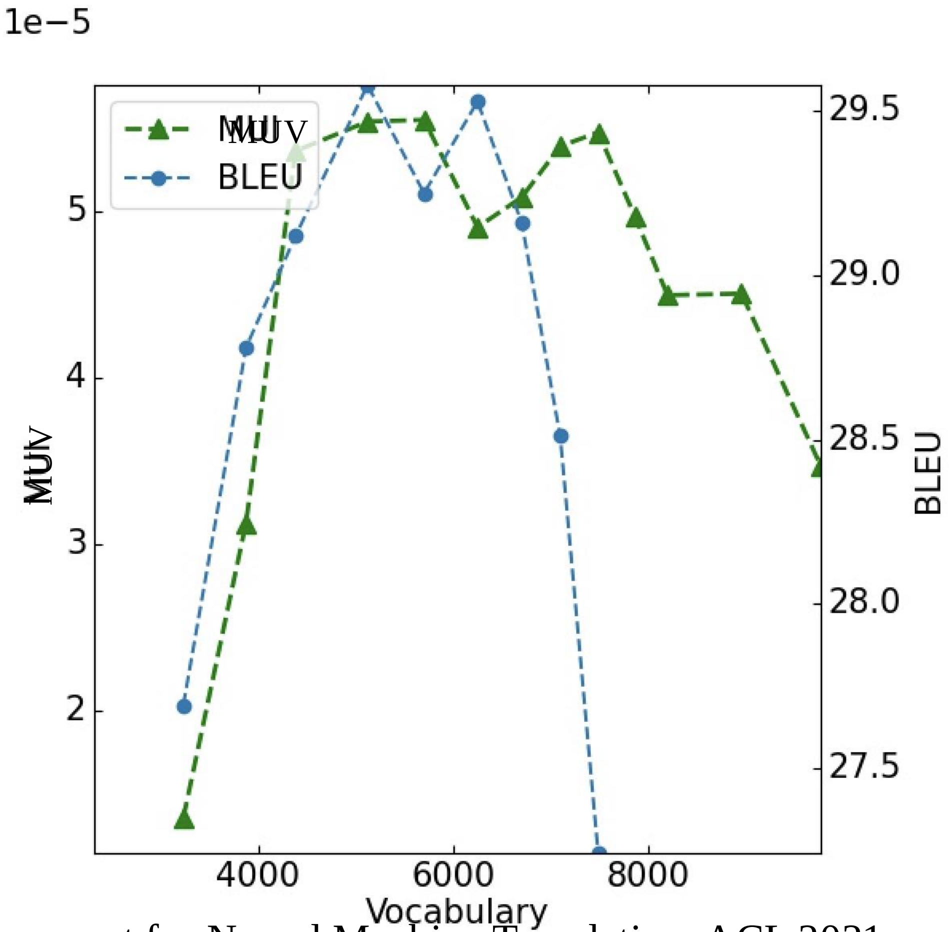


Xu, Zhou, Gan, Zheng, **Lei Li**. Vocabulary Learning via Optimal Transport for Neural Machine Translation. ACL 2021a

MUV Indicates MT Performance

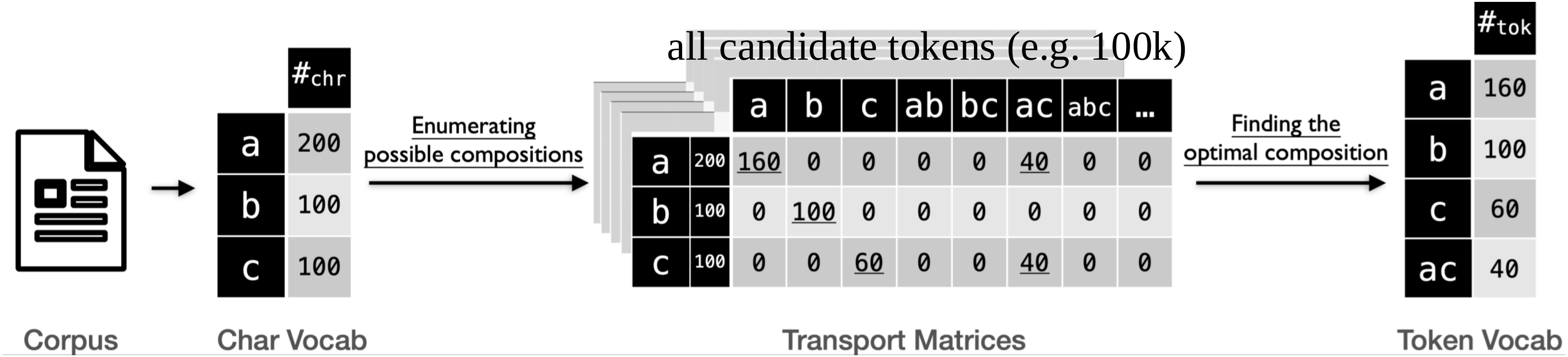
- MUV and BLEU are correlated on two-thirds of tasks
- A good coarse-grained evaluat





Xu, Zhou, Gan, Zheng, **Lei Li**. Vocabulary Learning via Optimal Transport for Neural Machine Translation. ACL 2021a

VOLT: Vocabulary Building via Transportation



- Maximizing MUV for vocabulary
  - $max - (H(V_{t+1}) - H(V_t))$
- Instead, maximizing the lower bound ==> Optimal Transport
  - $\max_t (maxH(V_t) - maxH(V_{t+1}))$

Reducing MUV Optimization to OT

- The vocabulary with the maximum MUV
  - Maximum gap between IPC of a vocabulary (with size t) and that of a smaller vocabulary (with size <t)
  - $max - (H(V_{t+1}) - H(V_t))$
- Intractable, instead to maximize lower-bound
- ==> $\max_{t}(maxH(V_t) - maxH(V_{t+1}))$
- Finding $\max_{v} H(v)$ ==> Optimal Transport

Xu, Zhou, Gan, Zheng, Li. Vocabulary Learning via Optimal Transport for Neural Machine Translation. ACL 2021.

VOLT: Finding the Optimal Vocabulary

- Entropy-regularized Optimal Transport

$$\min_{P\in\mathbb{R}^{m\times n}} \langle D, P\rangle - H(P)$$

subject to

$$\forall i \in Char, \sum_{j\in V_n} P_{i,j} = \hat{P}(i)$$

$$\forall j \in V_n, \Big|\sum_{i\in Char} P_{i,j} - \hat{P}(j)\Big| = \epsilon$$

- Sinkhorn's algorithm (from [Sinkhorn 1967])

Transportation matrix P

Char \ Tok	a	ab	bc
a	$P_{a,a}$	$P_{a,ab}$	$P_{a,bc}$
b	$P_{b,a}$	$P_{b,ab}$	$P_{b,bc}$
c	$P_{c,a}$	$P_{c,ab}$	$P_{c,bc}$

Cost matrix D

Char \ Tok	a	ab	bc
a	0	ln2	∞
b	∞	ln2	ln2
c	∞	∞	ln2

Xu, Zhou, Gan, Zheng, **Lei Li**. Vocabulary Learning via Optimal Transport for Neural Machine Translation. ACL 2021a.

Encoding and Decoding with VOLT

- VOLT uses a greedy strategy to encode text with a constructed sub-word level vocabulary similar to BPE.

- The vocabulary includes all basic characters.
  - To encode text, it first splits sentences into character-level tokens.
  - Then, we merge two consecutive tokens into one token if the merged one is in the vocabulary solved by OT.
  - This process keeps running until no tokens can be merged.
  - Out-of-vocabulary tokens will be split into smaller tokens.

Outline

- Subword tokenization: Byte-Pair-Encoding
  - Code walk through
- Information-theoretic vocabulary (VOLT)
- Practical Considerations in LLM
- Vocabulary sharing and impact on multilingual performance
- Tokenizer-free model (BLT)

Practical Consideration in LLMs

- deduplication
- sentencepiece
- Code (programming languages)
- Numbers
- multilingual

Corpus Deduplication and Filtering

- LLaMA 3 deduplicate at
  - url level dedup
  - document level dedup using minHash
  - line level dedup using SHA-1 64 bit hash code for every 30m docs
    - to remove boilerplate, e.g. navigation menu, cookie warning, contact info

- Filter
  - n-gram repeats in one line
  - “dirty word” counting
  - token distribution KL divergence too different from corpus

More Subword Tokenization

- BBPE: byte-level BPE (universal for all languages)

- Wordpiece:
  - like BPE
  - but instead of merge with most frequent pairs, merge a and b, if p(b|a) will be maximized

- SentencePiece:
  - Uniform way to treat space, punctuation
  - Use the raw sentence, replacing space ‘ ’ with _ (U+2581)
  - Then split character and do BPE

Kudo and Richardson, SentencePiece, 2018

Handling Code: Pre-tokenization

Using regular expression to split the sequences

GPT-4

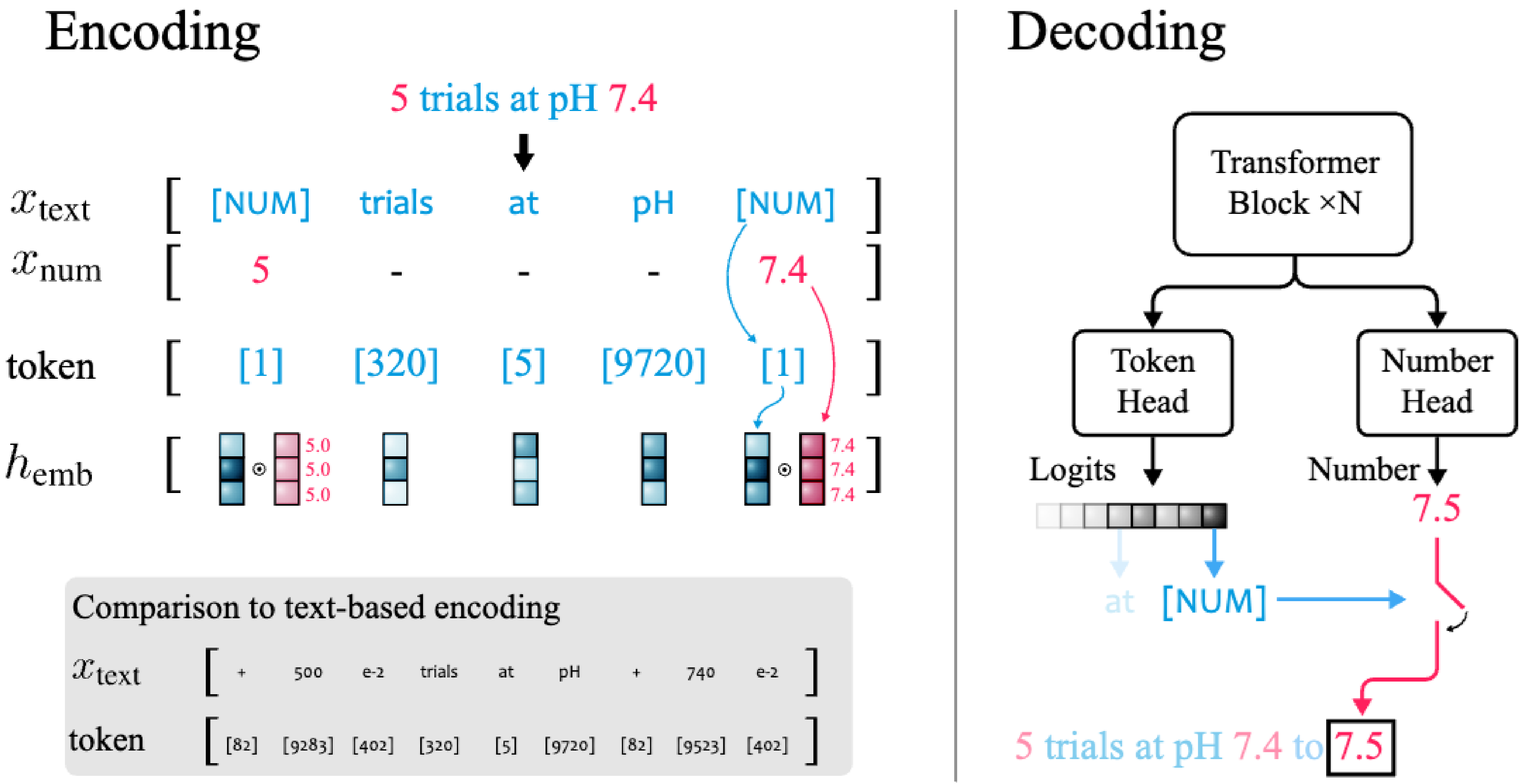
```
(?i:'s|'t|'re|'ve|'m|'ll|'d)|[^\r\n\p{L}\p{N}]?\p{L}+|\p{N}{1,3}| ?[^\s\p{L}\p{N}]+[\r\n]*|\s*[\r\n]+|s+(?!\S)|\s+
```

English contractions | Words | Digits | Non-alphanumeric | Line breaks | Trailing whitespace | whitespace

example: .append => a single token

Dagan et al. Getting the most out of your tokenizer for pre-training and domain adaptation. ICML 2024.

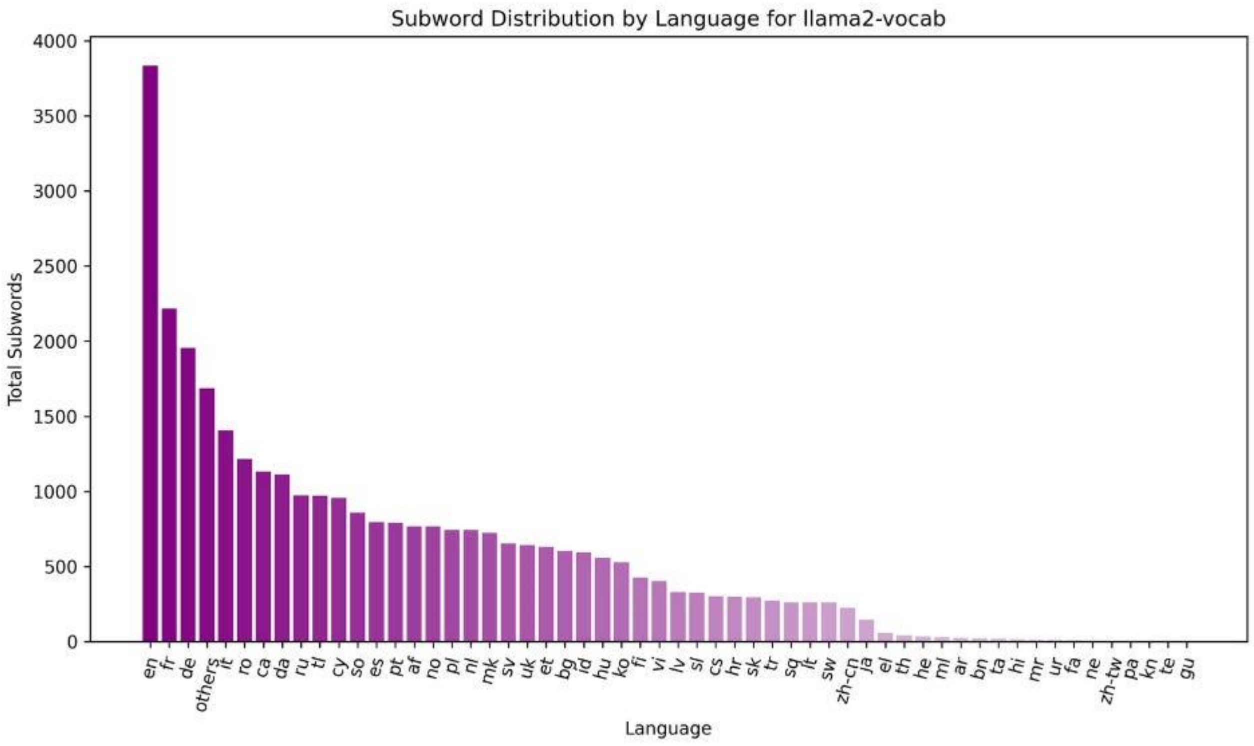
Handling numbers: Enable math in LLM

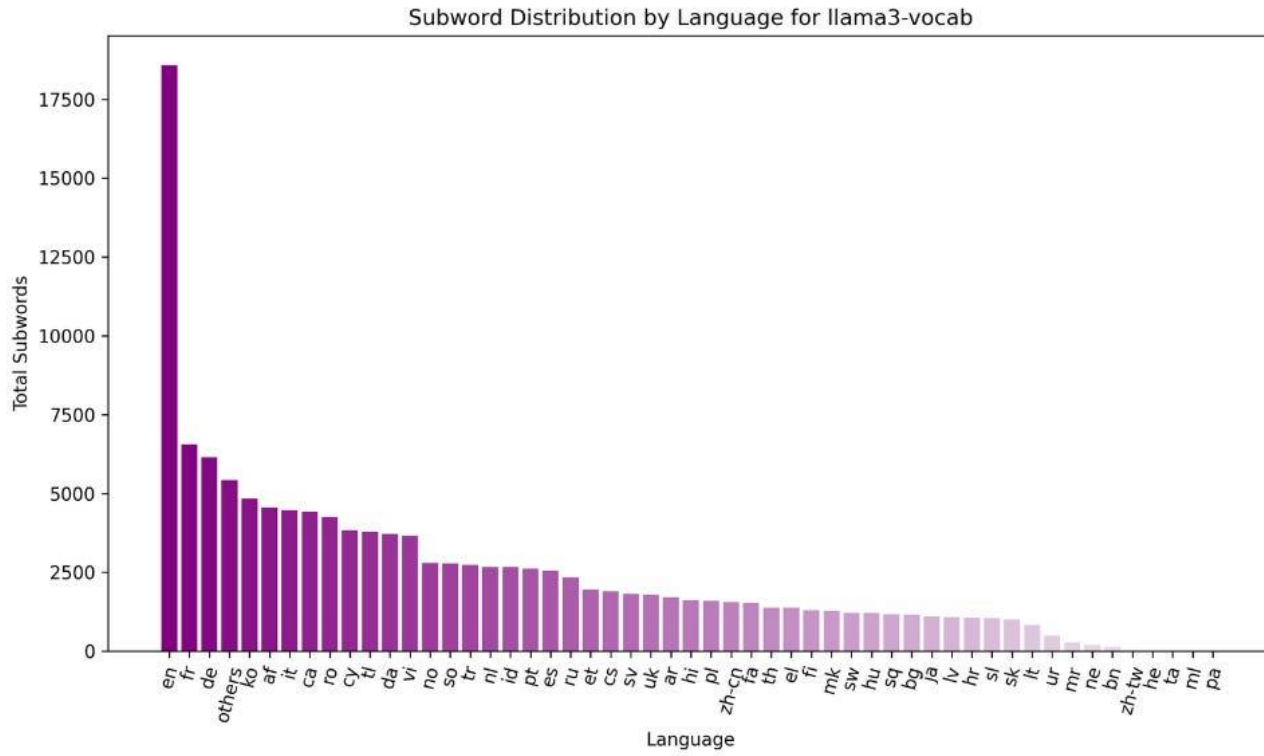


xVal: A Continuous Numerical Tokenization for Scientific Language Models. Golkar et al 2023.

LLaMA3’s multilingual vocabulary

- 32k (LLaMA 2) ➔ 128k tokens (LLaMA 3.1)
  - 100k from openAI’s tiktoken (from original 200k)
  - 28k allocated to multilingual





https://www.icodeformybhasa.com/p/exploring-multilingual-aspects-and

How to construct multilingual vocabulary?

- combining documents from multiple (176 in LLaMA3) languages, about 8% of total text, then apply BPE on the joint corpus
- obtain the same amount of BPE token for each language and then merge.
- allocating the capacity of each language by balancing the average log probability (ALP)

Zheng et al. Allocating large vocabulary capacity for crosslingual language model pre-training. EMNLP 2021.
Liang et al. XLM-V: Overcoming the Vocabulary Bottleneck in Multilingual Masked Language Models. EMNLP2023.

Demo

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

https://koala.sh/tools/free-gpt-tokenizer

Outline

- Subword tokenization: Byte-Pair-Encoding
  - Code walk through
- Information-theoretic vocabulary (VOLT)
- Practical Considerations in LLM
- Vocabulary sharing and impact on multilingual performance
- Tokenizer-free model (BLT)

Vocabulary Sharing

English: television	**Spanish:** televisión
French: television	**Italian:** television
Dutch: televisie	**Portuguese:** televisão
Swedish: television	**Finnish:** televisio

Embedding Finetuning for LLM

- Construct a small instruction-finetuning dataset using 10k bilingual parallel data
- Finetune LLaMA-7B
- Examine the translation performance of
  - The supervision bilingual direction (bilingual)
  - All other directions (multilingual)

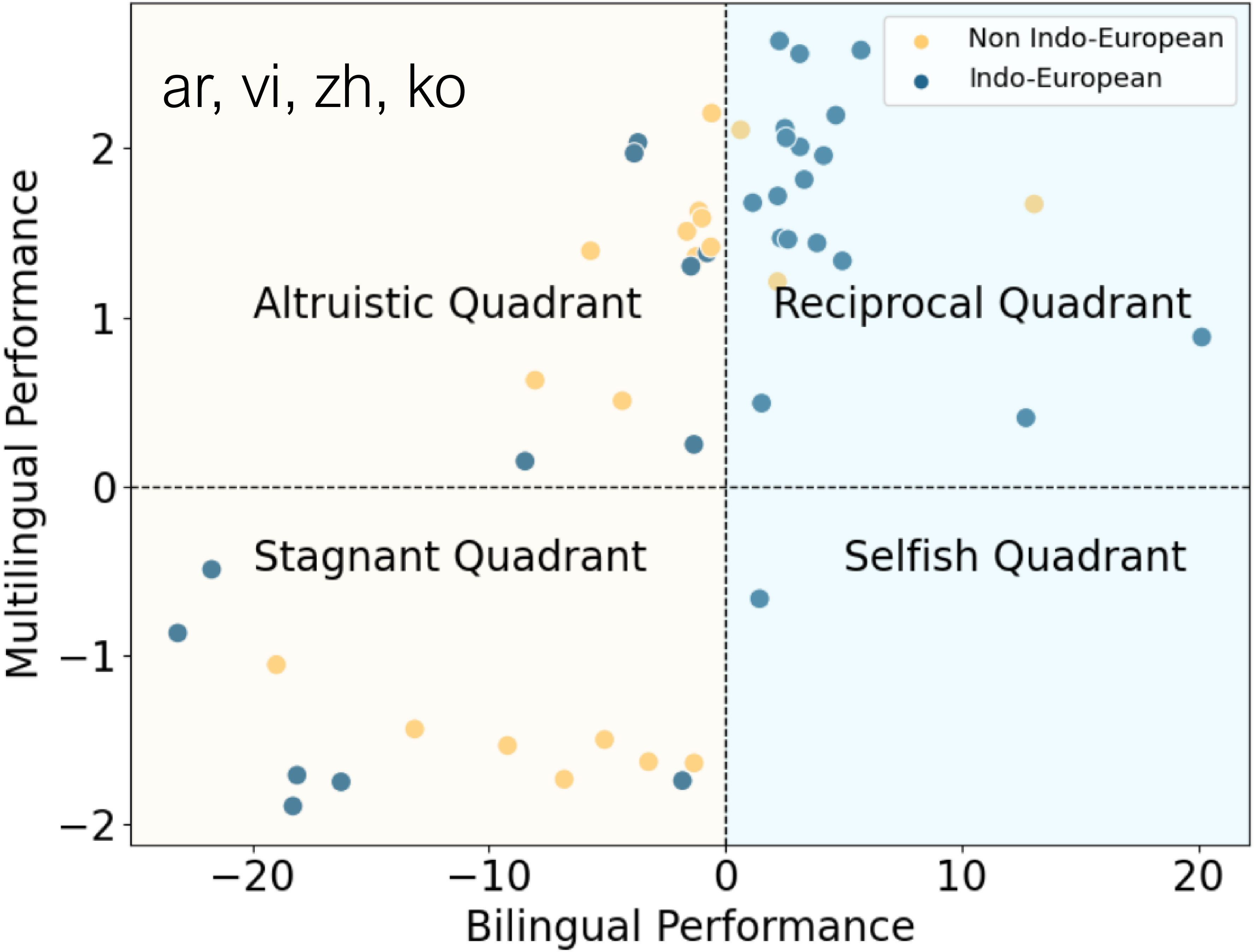
How Vocabulary Sharing Facilitates Multilingualism in LLaMA? Yuan et al, ACL

Does embedding FT promote bilingual & multilingual translation performance?

Quadrant	Performance		Case Languages
	Bilingual	Multilingual	
Reciprocal	↑	↑	cs, da, fr, de
Altruistic	↓	↑	**ar, vi, zh, ko**
Stagnant	↓	↓	**Km, lo, gu, te**
Selfish	↑	↓	**hi**

How Vocabulary Sharing Facilitates Multilingualism in LLaMA? Yuan et al, ACL 2024

Fine-tuning on bilingual data does not always bring benefits to supervised direction!



How Vocabulary Sharing Facilitates Multilingualism in LLaMA? Yuan et al, ACL 2024

Stagnant Quadrant – Over-tokenization

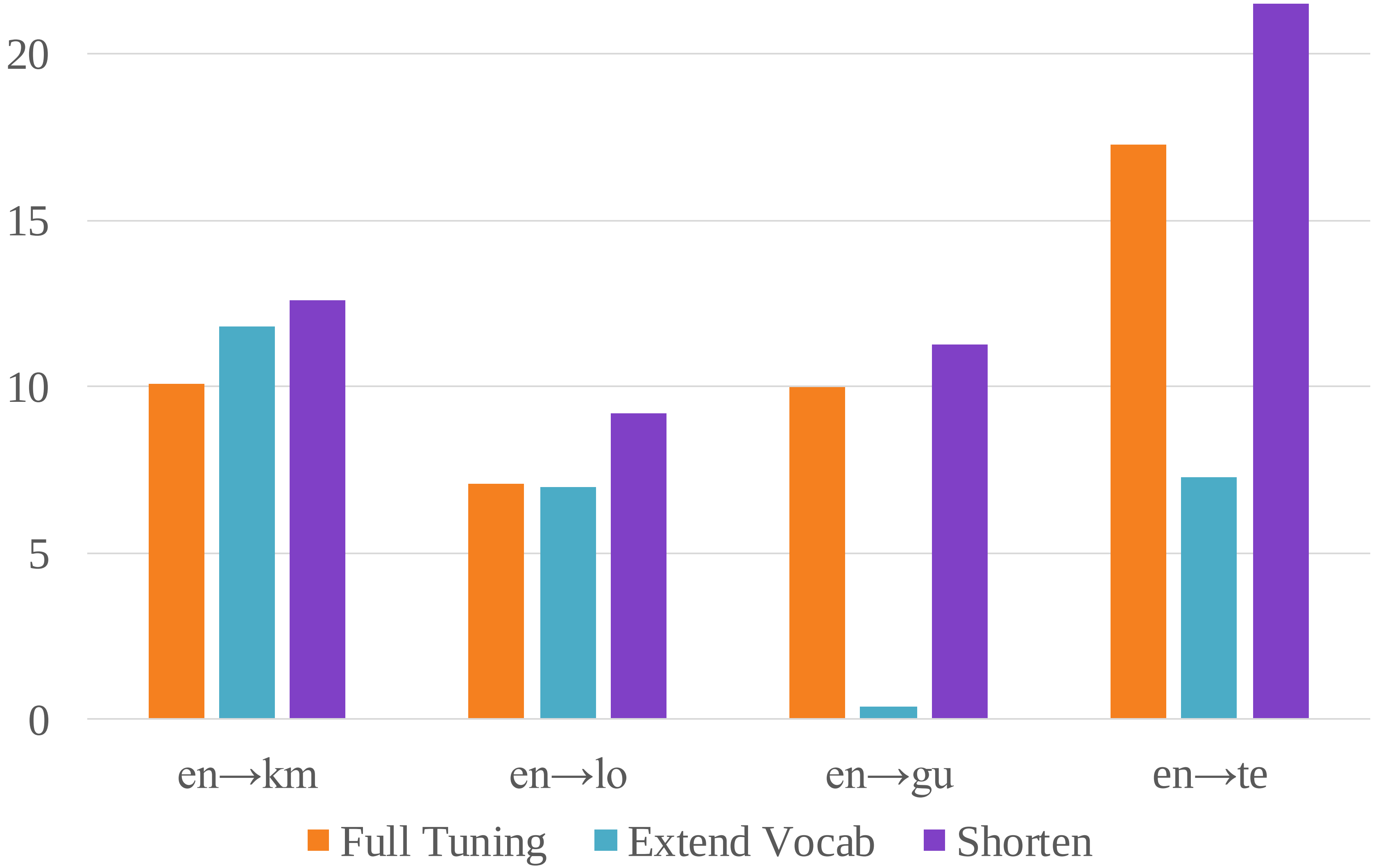
- Byte-BPE (BBPE) produces longer byte level token sequence than the number of characters

- 饕 [tāo] (gluttonous) ➔ three tokens [227, 234, 260]

- Implication for improvement:
  - shortening: remove the common prefix 227

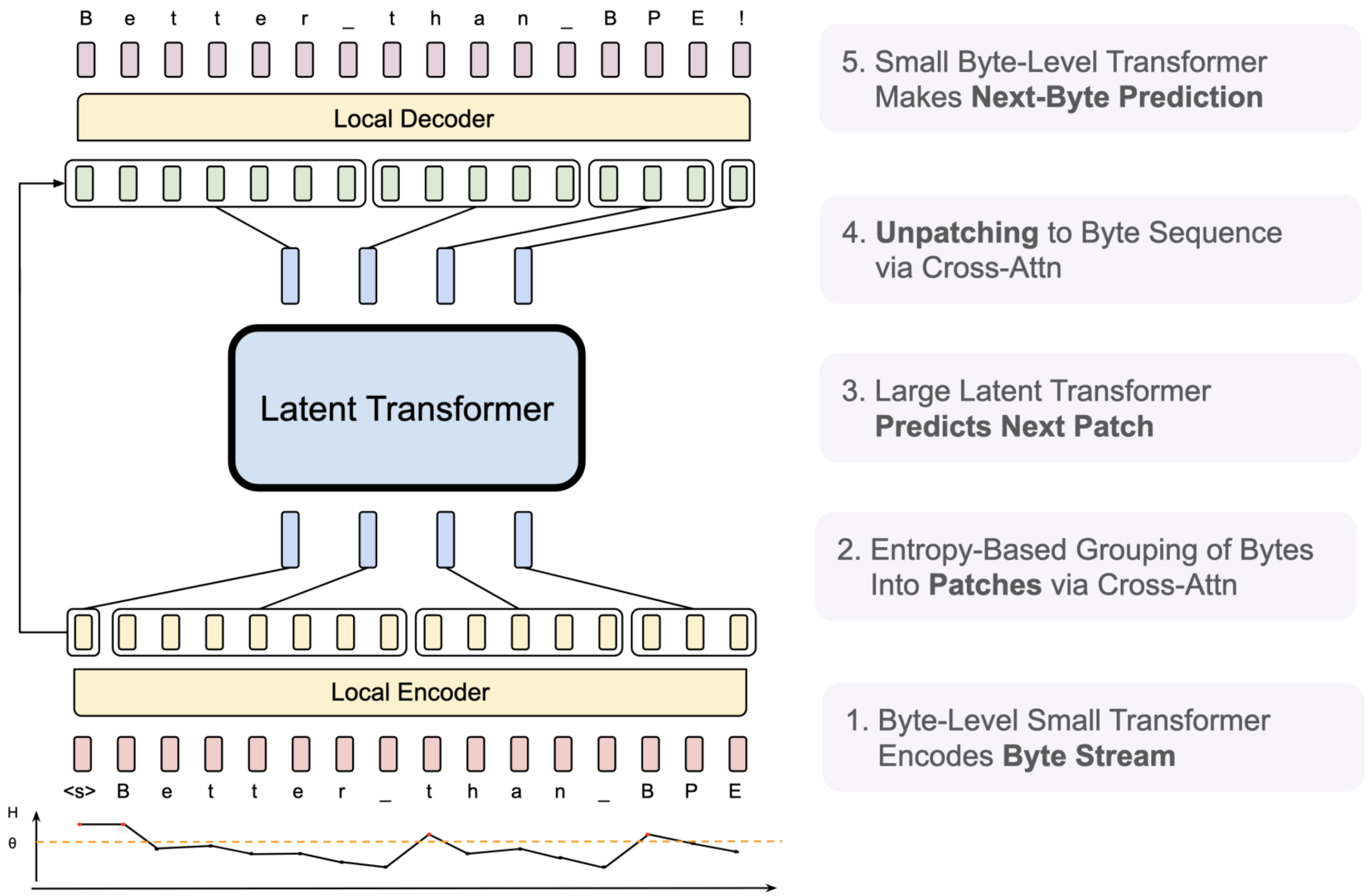
How Vocabulary Sharing Facilitates Multilingualism in LLaMA? Yuan et al, ACL 2024

Stagnant Quadrant: expanding vocab 👎 shortening 👍



How Vocabulary Sharing Facilitates Multilingualism in LLaMA? Yuan et al, ACL 2024

Tokenizer-free Model – Byte Latent Transformer



Pagnoni et al. Byte Latent Transformer: Patches Scale Better Than Tokens. 2024.

Summary

- Subword tokenization: Byte-Pair-Encoding
  - iteratively merging most frequent pairs of tokens
- Information-theoretic vocabulary (VOLT)
  - solving entropy constrained optimal transport problem
- Pre-tokenization through regex
- Number treatment
- Vocab sharing impact multilingual performance
  - how to solve languages in stagnant quad

Quiz 5

https://canvas.cmu.edu/courses/44373/quizzes/140013