A/ Part/ition/ Cover/ Approach/ to/ Token/ization

Jia Peng Lim
Singapore Management University

Shawn Tan
MIT-IBM Watson AI Lab

Davin Choo*
Harvard University
* Equal advising

Hady Lauw*
Singapore Management University
* Equal advising

Contributions

NEURAL INFORMATION PROCESSING SYSTEMS

1. An intuitive proof that tokenization is NP-hard via a reduction from Vertex Cover

2. Partition cover formulation, independent of merge patterns

3. Proposed GreedTok, a novel base text tokenization algorithm with better compression.

4. Empirical approximation of objective.

5. Downstream language pretraining comparison.

Tokenization is NP-Hard

Finding: An optimal solution for *Tok* decision problem is the solution for vertex cover and vice versa for vertex cover problem instances.

Tokenization search problem (*Tok*)

Given

- Fixed alphabet $\boldsymbol{\Sigma}$
- Base token set $\mathbf{B} = \{(w) : w \in \boldsymbol{\Sigma}\}$ as all singleton characters
- Corpus $\mathscr{C} = (\mathbf{W}, \mathrm{COUNT})$
- Token budget $k \in \mathbb{N} > 0$
- Set of candidate tokens $\mathbf{T} \subseteq \boldsymbol{\Sigma}^+$

Goal: Find a subset $\mathbf{S} \subseteq \mathbf{T}$ such that $|\mathbf{S}| \leq k$ and minimise

$$\sum_{W \in \mathbf{W}} \mathrm{PARTITION}(\mathrm{W}, \mathbf{S} \cup \mathbf{B}) \cdot \mathrm{COUNT}(W).$$

Tokenization decision problem: given integer threshold $\mathscr{L} \in \mathbb{N} > 0$

Determine whether there exists a subset $\mathbf{S} \subseteq \mathbf{T}$ such that $|\mathbf{S}| \leq k$ and

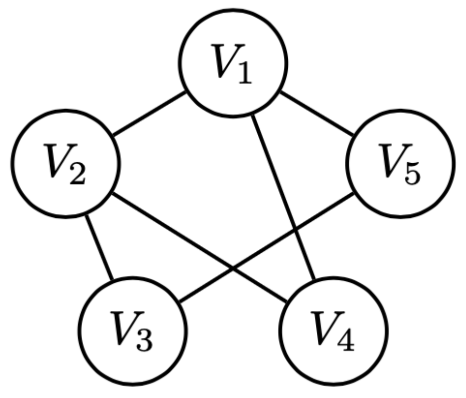
$$\sum_{W \in \mathbf{W}} \mathrm{PARTITION}(\mathrm{W}, \boxed{\mathbf{S} \cup \mathbf{B})} \cdot \mathrm{COUNT}(W) \leq \mathscr{L}.$$

Differentiate between singletons $\mathbf{B}$ and selected candidate tokens $\mathbf{S}$

Representing strings as graphs, given:

- Arbitrary vertex cover graph $\mathscr{G} = (\mathbf{V}, \mathbf{E})$.
- Set of words as $\mathbf{W} = \{W_{i,j} : V_i, V_j \in \mathbf{E}\}$
- Each word $W_{i,j} = (@, V_i, @, V_j, @)$, with a count of 1
- Set of candidate tokens as $\mathbf{T} = \{(@, V_i, @) : V_i \in \mathbf{V}\}$
- Set $\mathscr{L} = 3|\mathbf{W}| = 3|\mathbf{E}|$ and vertex cover's k to *Tok*'s k.

Notice that every edge ⇔ word and every vertex ⇔ token



$$\begin{aligned}
\boldsymbol{\Sigma} &= \{V_1, V_2, V_3, V_4, V_5, @\} \\
\mathbf{B} &= \{(V_1), (V_2), (V_3), (V_4), (V_5), (@)\} \\
\mathbf{W} &= \{(\underline{@, V_1, @}, V_2, @), (\underline{@, V_1, @}, V_4, @), (\underline{@, V_1, @}, V_5, @), \\
&\qquad (@, V_2, \underline{@, V_3, @}), (@, V_2, \underline{@, V_4, @}), (\underline{@, V_3, @}, V_5, @)\} \\
\textsc{count}(W) &= 1, \quad \forall W \in \mathbf{W} \\
\mathbf{T} &= \{\underline{(@, V_1, @)}, (@, V_2, @), \underline{(@, V_3, @)}, \underline{(@, V_4, @)}, (@, V_5, @)\} \\
k &= k \\
\ell &= 3|\mathbf{W}| = 3|\mathbf{E}| = 18
\end{aligned}$$

An example tokenization problem instance. The tokens corresponding to the vertex cover $\mathbf{S} = \{V_1, V_3, V_4\}$ are underlined in $\mathbf{T}$. A possible tokenization of $\mathbf{W}$ using $\mathbf{S} \cup \mathbf{B}$ is also given with tokens in $\mathbf{S}$ being underlined, showing that each word in $\mathbf{W}$ only needs 3 tokens.

NEURAL INFORMATION PROCESSING SYSTEMS

GreedTok: Our greedy tokenizer

Algorithm design: from partition to covers

Cover Pairs and not singletons

Tokens

T1 : BA

T2 : NA

Example: B A N A N A

As pairs: T1 T2 T2 ✓

We set constraints such that the ends of the cover must be uncovered to demarcate valid covers/separations.

As singles: T1 T1 T2 T2 T2 T2 ?

Covering as singles introduces additional complications when verifying whether a cover is valid for updating scores.

Minimizing partitions is maximizing covers

$$\text{Length of word} = \text{\# covers} + \text{\# partitions} = 3 + 3 = 6$$

Relax MIP (Min. Partition) → Max. Cover with constraints → Greed tokens for potential covers

GreedTok: Our greedy tokenizer

NEURAL INFORMATION PROCESSING SYSTEMS

Design comparisons

Base Text Tokenizers

	BPE	Unigram	GreedTok
Starting Tokens	Singletons	All possible	All possible
Token Selection	Greed *k*	Eliminate all but *k*	Greed *k*
Objective	Max pairwise merges between selected tokens	Max subword probabilities	Max valid singleton-pair covers
Solving Complexity	$O(k \cdot \sum_{W \in \mathbf{W}} \lvert W \rvert)$	$O(\lvert \mathbf{T} \rvert \cdot \log k \cdot \sum_{W \in \mathbf{W}} \lvert W \rvert)$	$O(\lvert \mathbf{T} \rvert \cdot \boxed{\cancel{k}} \cdot \sum_{W \in \mathbf{W}} \lvert W \rvert)^{*}$
Encoding Complexity	$O(\lvert W \rvert^2)$	$O(k \lvert W \rvert)$	$O(\lvert W \rvert^2 \log W)$

Evaluating GreedTok's Compression

Finding: GreedTok has better compression than BPE and Unigram.

Dataset	$\lvert\mathbf{W}\rvert$	$\sum_{W}^{\mathbf{W}} c_W$	$\lvert\mathbf{T}\rvert$	max $\lvert\mathbf{S}\rvert$	Time
UN	105K	37M	884K	5K	6s
arχiv	881K	366M	7,626K	5K	63s
wiki	8,769K	2,949M	93.5M	10K	11m
PubMed	6,527K	4,149M	121M	10K	11m

Table 1 in paper

These corpora contains **100K to 8M unique space-delimited words**, with of **800K to 121M token candidates**. The runtime of GreedTok is competitive, with the longest duration being **11 minutes** on AMD EPYC 9654 @ 2.40GHz.

GreedTok (GTK) outperforms BPE and Unigram in larger corpora (arXiv, PubMed, wiki), with mean improvement of **2.88% over BPE** and **3.43% over UNIGRAM**.

	k	1000	2000	3000	4000	5000		2000	4000	6000	8000	10000
GTK Tokens/Word	UN	1.607	1.374	1.268	1.205	1.163	PubMed	1.603	1.397	1.301	1.244	1.206
BPE Tokens/Word	UN	1.688	1.431	1.311	1.241	1.194	PubMed	1.650	1.431	1.328	1.266	1.225
GTK's Improvement (%)	UN	**4.86**	**3.99**	**3.33**	**2.92**	**2.54**	PubMed	**2.85**	**2.38**	**2.02**	**1.75**	**1.52**
UNIGRAM Tokens/Word	UN	1.655	1.385	1.261	1.193	1.148	PubMed	1.699	1.465	1.359	1.297	1.257
GTK's Improvement (%)	UN	**2.90**	**0.78**	-0.51	-0.97	-1.30	PubMed	**5.63**	**4.63**	**4.21**	**4.05**	**4.02**
GTK Tokens/Word	arχiv	1.742	1.475	1.349	1.275	1.226	wiki	1.692	1.489	1.389	1.326	1.283
BPE Tokens/Word	arχiv	1.837	1.551	1.407	1.320	1.263	wiki	1.731	1.519	1.413	1.347	1.301
GTK's Improvement (%)	arχiv	**5.12**	**4.94**	**4.15**	**3.41**	**2.89**	wiki	**2.26**	**1.98**	**1.71**	**1.53**	**1.37**
UNIGRAM Tokens/Word	arχiv	1.793	1.558	1.444	1.378	1.332	wiki	1.793	1.558	1.444	1.378	1.332
GTK's Improvement (%)	arχiv	**6.74**	**4.92**	**4.30**	**3.96**	**3.88**	wiki	**5.62**	**4.43**	**3.84**	**3.75**	**3.70**

Table 2 in paper

NEURAL INFORMATION PROCESSING SYSTEMS

Evaluating GreedTok's Approximability

We can empirically show that GreedTok is $d_{inst}(1 - 1/e)$!

Maximum Coverage Greedy is a $1 - 1/e$ algorithm, best for Maximum Coverage Problem.

However, *Tok* is *neither submodular nor supermodular*; uncertain if GreedTok's approximation ratio is $1 - 1/e$; see Appendix A

Weighted Maximum Coverage's Greedy (GreedWMC) is GreedTok's upper bound.

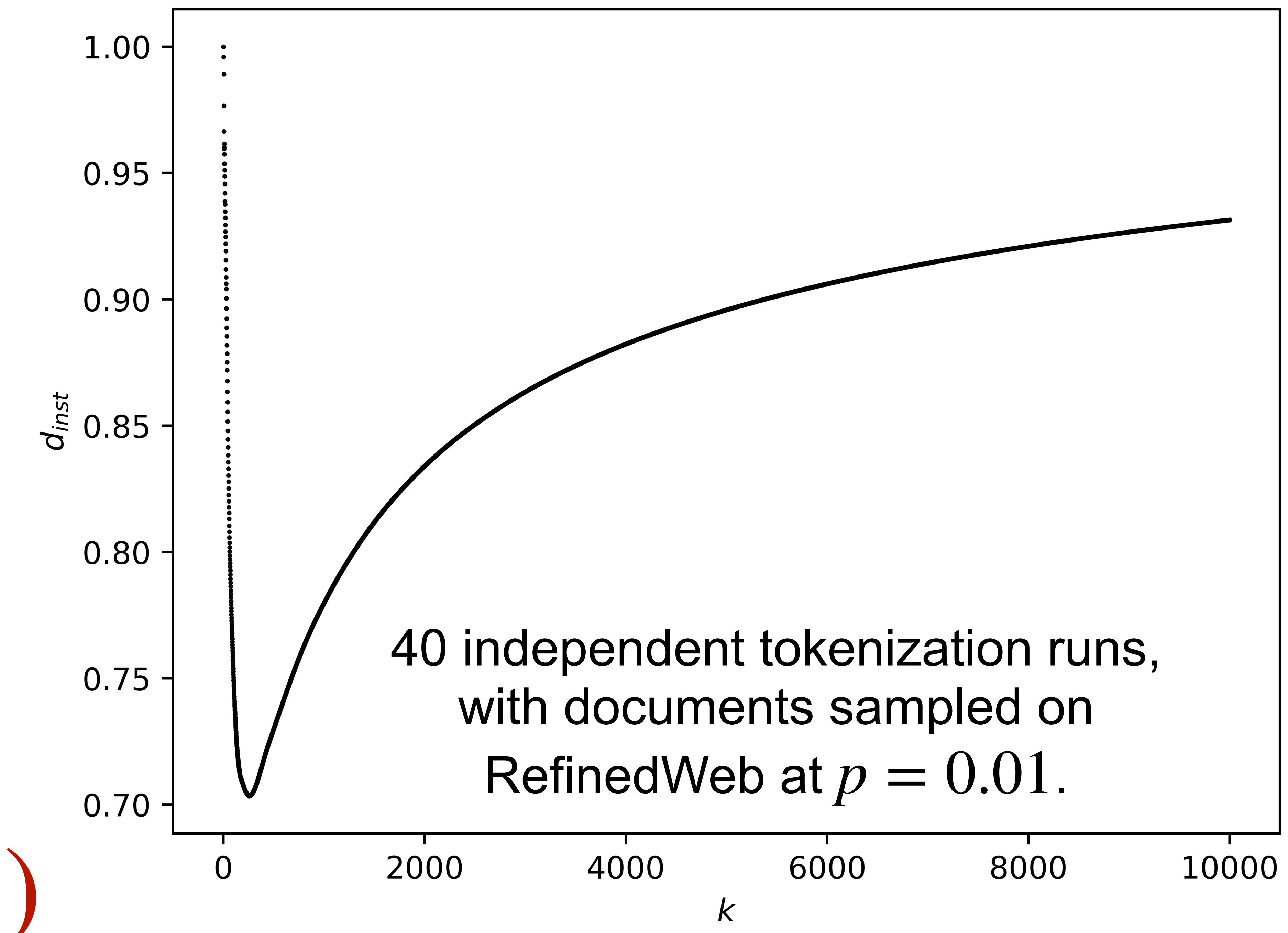
Score: # cover in *instance*

Optimal score for WMC $\geq$ Optimal for *Tok*

GreedWMC score

GreedTok score

$$d_{inst} = \frac{GreedTok}{GreedWMC}$$



For large *k*, GreedTok is $0.9(1 - 1/e)$

Evaluating GreedTok's Language Pretraining

NEURAL INFORMATION PROCESSING SYSTEMS

Experiment Design: GreedTok vs BPE, 1B-decoder-only Transformers, token sets 75% similar

The token count statistics for all three settings. GreedTok uses nearly 18% fewer tokens to represent the entire DCLM Dedup dataset. The total training tokens used is around 629B tokens. Dataset % differences due to tokenised compression and checkpointing.

Experiment Name	Tokenizer	Full dataset tokens	Training tokens	Dataset %
BPEM	BPE	$8.94 \cdot 10^{11}$	$6.29 \cdot 10^{11}$	70.35%
Equal Tokens (GTET)	GREEDTOK	$7.35 \cdot 10^{11}$	$6.29 \cdot 10^{11}$	85.58%
Equal Proportion (GTEP)	GREEDTOK	$7.35 \cdot 10^{11}$	$5.03 \cdot 10^{11}$	68.47%

Table 3 in paper

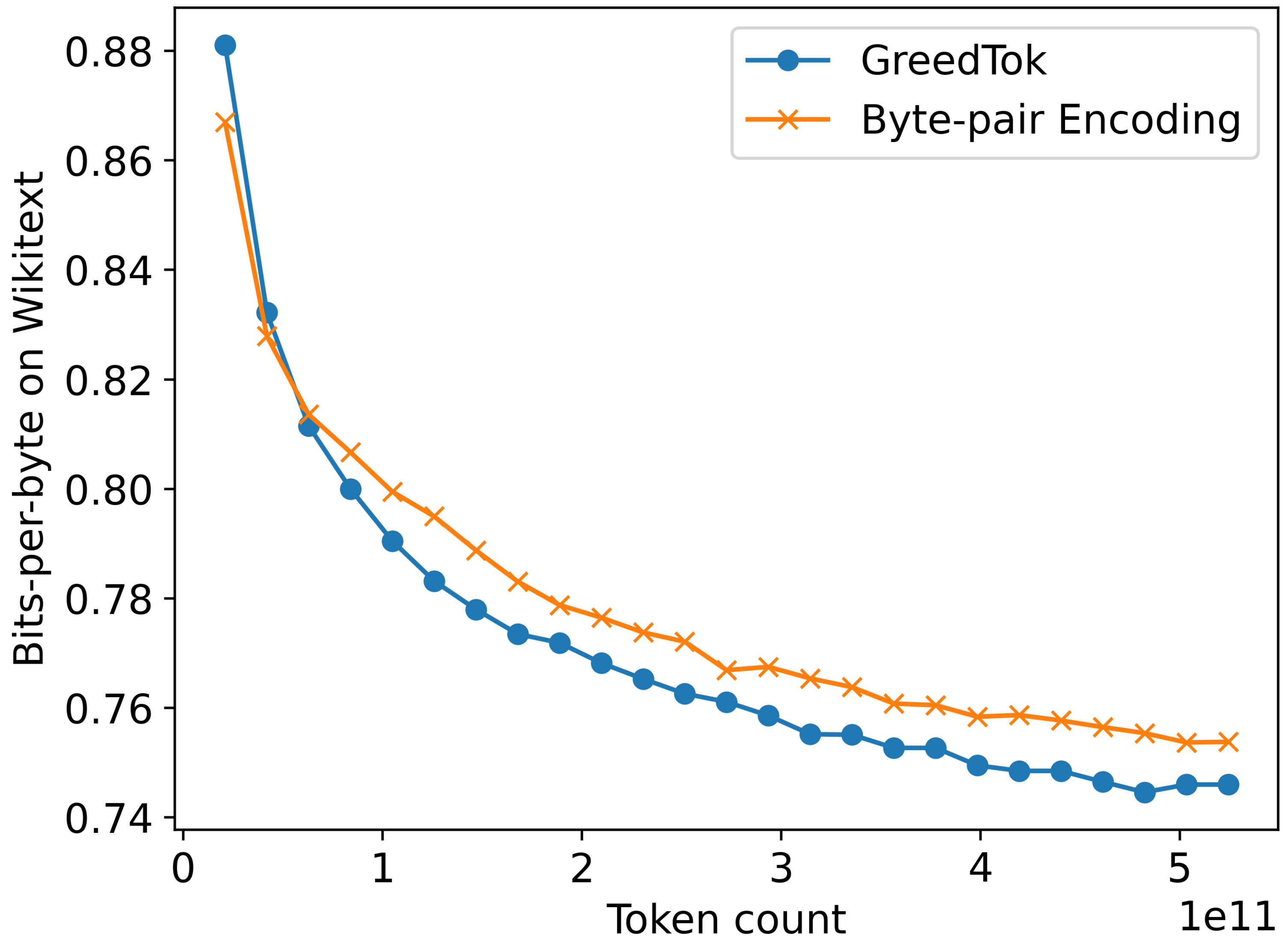
Our two GreedTok settings:

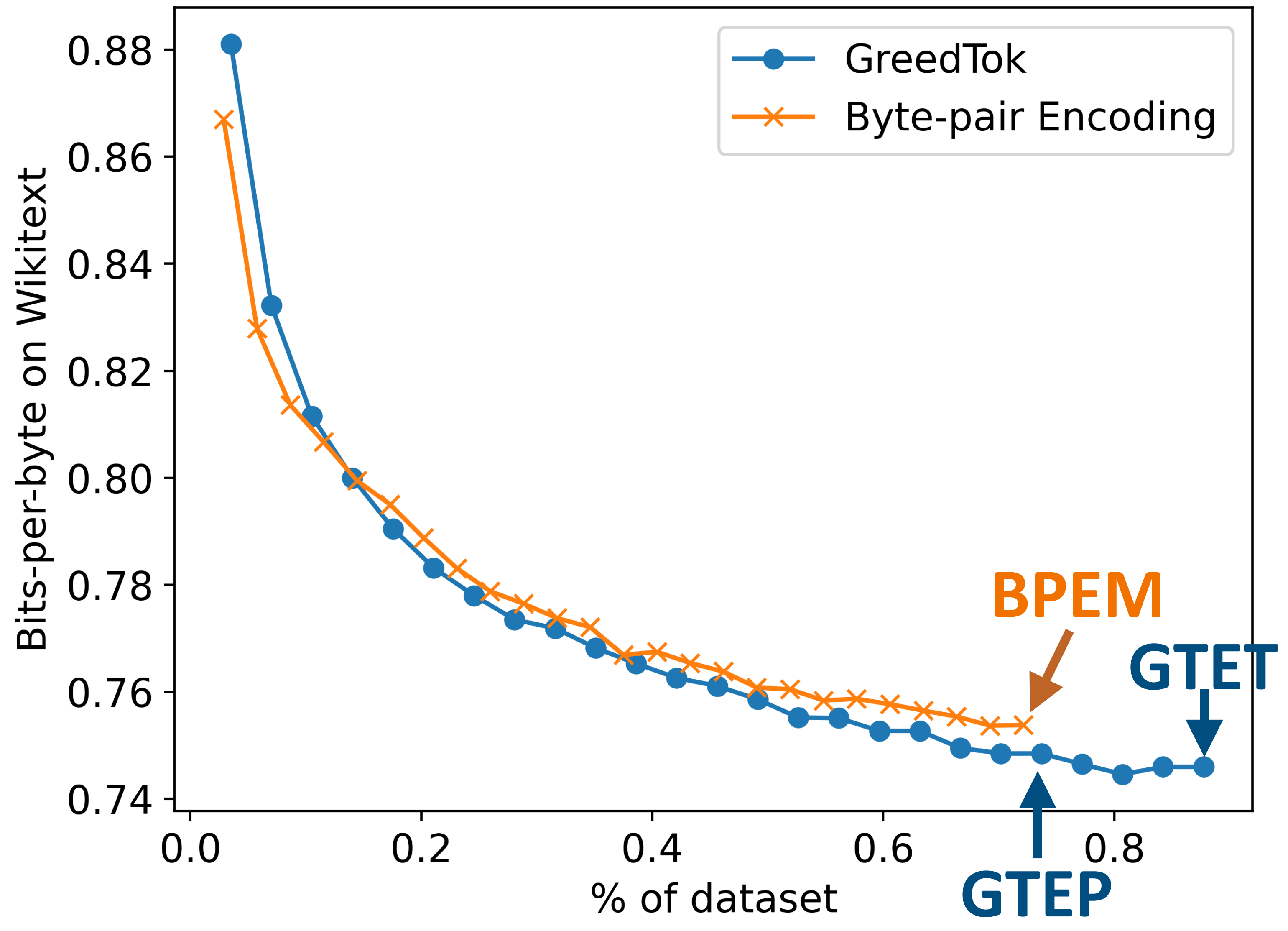
- **GTET**: trains on an equal amount of tokens compared to BPEM
- **GTEP**: trains on an comparable amount of text coverage compared to BPEM, note that this results in GTEP training on fewer tokens due to higher compression

Evaluating GreedTok's Language Pretraining

NEURAL INFORMATION PROCESSING SYSTEMS

Results: GreedTok is ahead/comparable, on common benchmarks and bits/bytes.





We plot the bits/byte improvement across phase 1 training for model using GreedTok and BPE on different scales. The bits/byte metric is independent of tokenization and reflects true compression performance on the underlying data. Since both GTET and GTEP are equivalent in phase 1 for the first 100,000 steps, we examine bits/byte improvement on Wikitext with different scales on the x-axes.

	Accuracy (normalized)						*Accuracy*						*bits/byte*
	ARC-c	ARC-e	Hella-Swag	OBQA	PIQA	SciQ	BoolQ	COPA	LAMB-BADA	RACE	Wino-grande	*Avg.*	Wikitext
BPEM	36.2	67.9	65.6	40.0	75.7	89.8	65.8	81.0	61.1	36.4	62.8	62.0	0.7066
GTEP	37.6	68.8	64.9	39.6	75.6	90.0	67.6	79.0	63.9	36.8	**63.5**	62.5	0.7028
GTET	**38.3**	**70.0**	**65.7**	**40.6**	**75.8**	**90.5**	**67.7**	**82.0**	**64.6**	**37.7**	62.6	**63.2**	**0.6989**

Thanks!

Looking for industry positions!

Looking for academic positions!

Jia Peng Lim
Singapore Management University

Shawn Tan
MIT-IBM Watson AI Lab

Davin Choo*
Harvard University
* Equal advising

Hady Lauw*
Singapore Management University
* Equal advising

arXiv

Github