Entropy-guided Vocabulary Augmentation of Multilingual Language Models for Low-resource Tasks

Arijit Nag
IIT Kharagpur
arijitnag@iitkgp.ac.in

Bidisha Samanta
IIT Kharagpur
bidisha@iitkgp.ac.in

Animesh Mukherjee
IIT Kharagpur
animeshm@cse.iitkgp.ac.in

Niloy Ganguly
IIT Kharagpur
niloy@cse.iitkgp.ac.in

Soumen Chakrabarti
IIT Bombay
soumen@cse.iitb.ac.in

Abstract

Multilingual language models (MLLMs) like mBERT promise to extend the benefits of NLP research to low-resource languages (LRLs). However, LRL words are under-represented in the wordpiece/subword vocabularies of MLLMs. This leads to many LRL words getting replaced by UNK, or concatenated from morphologically unrelated wordpieces, leading to low task accuracy. (Pre)-training MLLMs after including LRL documents is resource-intensive in terms of both human inputs and computational resources. In response, we propose EVALM (entropy-based vocabulary augmented language model), which uses a new task-cognizant measurement to detect the most vulnerable LRL words, whose wordpiece segmentations are undesirable. EVALM then provides reasonable initializations of their embeddings, followed by limited fine-tuning using the small LRL task corpus. Our experiments show significant performance improvements and also some surprising limits to such vocabulary augmentation strategies in various classification tasks for multiple diverse LRLs, as well as code-mixed texts. We will release the code and data to enable further research[1].

1 Introduction

It is common practice to start with a multilingual language model (MLLM) like mBERT[2] or XLM-R (Conneau et al., 2020), which has been pre-trained with large multilingual corpora, and fine-tune the MLLM for diverse downstream tasks. Although MLLMs support many low-resource languages (LRLs), closer inspection of these MLLMs reveals that the portion of vocabulary allotted to LRLs can be orders of magnitude smaller than that allotted to high-resource languages (HRLs) such as English (Table 1).

Language	Vocab count	Percentage (%)
Bengali	946	0.79
Hindi	1852	1.55
Gujarati	404	0.34
Kannada	653	0.55
Malayalam	565	0.47
Tamil	832	0.7
Telugu	887	0.74
English*	64529–78984	53.98–66.07

Table 1: Representation of the vocabulary of various Indian languages in mBERT's wordpiece dictionary. *Based on basic to extended Latin script Unicode range.

Due to this imbalance, sometimes an LRL word may not be possible to segment into wordpieces as per the MLLM vocabulary, leading to the LRL word being conflated with the UNK (unknown) token. An even more insidious situation is that the MLLM vocabulary has enough (over-fragmented) wordpieces to assemble almost *any* LRL word (thereby dodging the obvious UNK alert), but the embeddings of these wordpieces collide with unrelated usage in HRLs, and/or are so sparsely trained that contextual aggregations fail to yield satisfactory LRL word embeddings which may lead to poor LRL task performance. On the other hand, significant human and computational investments are needed to create task-specific LRL corpora that are large enough to augment and retrain the MLLM vocabulary.

In this work, we address the setting where a MLLM (that is presumably deficient in LRL coverage) must be minimally fine-tuned after modest modification to its wordpiece vocabulary, guided by specific LRL tasks. We design a measure of damage to an LRL word, caused by wordpiece fragmentation, based on a suitably defined notion of entropy of the word and constituent wordpieces, with respect to the LRL task. This measure then guides the selection of LRL words with which the vocabulary should be augmented. Subsequently, we propose various ways to initialize the embeddings of these newly-introduced words, including

[1] https://github.com/NLPatCNERG/EVALM

[2] https://github.com/google-research/bert/blob/master/multilingual.md

Findings of the Association for Computational Linguistics: ACL 2023, pages 8619–8629
July 9-14, 2023

using information from the LRL itself, to 'importing' information from HRLs. We call the resulting system EVALM (entropy-based vocabulary augmented language model).

We study the effect of EVALM on an existing MLLM during the fine-tuning stage for various downstream classification tasks covering multiple LRLs and also a code-mixed language. Our study shows that, for most of the datasets, EVALM's vocabulary augmentation strategy helps improve LRL task performance by greater margins than recent best practices (Hong et al., 2021; Hofmann et al., 2022). A detailed analysis of successes and failures delineates the perimeter of EVALM's capabilities and guides our design choices.

2 Related Work

Continued pre-training (Tai et al., 2020; Ebrahimi and Kann, 2021; Wang et al., 2020; Chau et al., 2020) with or without vocabulary augmentation of existing LMs like monolingual BERT, multilingual BERT (mBERT), XLM-R, etc., proves beneficial for improving domain and language-specific performances over various tasks. Some works (Ruzzetti et al., 2021; Yu et al., 2021) focus on rare/OOV words. Liu et al. (2021) propose an embedding generator module in the pretrain-finetune pipeline to resolve vocabulary gaps. Adaptors (Sachidananda et al., 2021; Moon and Okazaki, 2020; Hofmann et al., 2021) are also showing promising outcomes in LRL modeling. Chung et al. (2020) explore multilingual vocabulary generation from language clusters. Minixhofer et al. (2021) transfer English LMs to new languages without expensive computation. Hofmann et al. (2022) propose a simple algorithm which modifies the tokenization process to preserve the morphological structure of a word. Others (Wang et al., 2019; Hong et al., 2021) focus on embedding initialization for newly added vocabulary words which are word fragments, which is also among our concerns.

3 Our system: EVALM

EVALM has three key components. The purpose of the first component (Section 3.1) is to identify (based on only the train fold) a subset of *vulnerable* LRL words whose assembly from wordpieces is likely to distort the embedding information made available to LRL labeling tasks. The second component (Section 3.2) comprises various possible policies to initialize the embeddings of the newly-introduced LRL words. In the third component, as in AVocaDo (Hong et al., 2021), we prevent overfitting to a small LRL task corpus by regularizing embeddings of corresponding wordpieces of each sentence obtained by the pre- and post-augmentation MLLM tokenizers.

Algorithm 1 LRL vocabulary selection.

Inputs:
- C-class LRL task training corpus D,
- MLLM tokenizer $\mathcal{T}$
- word frequency threshold θ
- entropy reduction threshold γ
- maximum size of augmentation set V_{new}

$W \leftarrow$ all words from corpus D
$S \leftarrow \bigcup_{w \in W} \mathcal{T}(w)$
compute $n(w, c)$ for all LRL words $w \in W, c \in C$
compute $n(s, c)$ for all wordpieces $s \in S, c \in C$
compute $p(c|w), p(c|s), H(w), H(s)$ as described
candidates $= \varnothing$
for each LRL word $w \in W$ **do**
    compute average wordpiece entropy $H_S(w) = \sum_{s \in \mathcal{T}(w)} H(s)/|\mathcal{T}(w)|$
    compute word frequency $n(w) = \sum_c n(w, c)$
    compute $\Delta_H(w) = \dfrac{H_S(w) - H(w)}{H_S(w)}$
    features of w are $\langle n(w), |\mathcal{T}(w)|, \Delta_H(w) \rangle$
    if $n(w) \geq \theta$ and $\Delta_H(w) \geq \gamma$ **then**
        add the feature triple to *candidates*
sort *candidates* in decreasing Δ_H

Output: Prefix of *candidates* of specified size as V_{new}

3.1 Vulnerable LRL word selection

We need a computationally efficient, task-sensitive surrogate of the value of introducing an LRL word into the wordpiece vocabulary. (Here we augment the vocabulary with whole LRL words, blocking their fragmentation entirely. More clever sharing of fragments is left for future work.)

Suppose LRL word w is not in the MLLM vocabulary; w is fragmented into wordpiece sequence $\mathcal{T}(w) = s_1, \ldots, s_T$ by the MLLM tokenizer $\mathcal{T}$. The LRL task has C class labels. A specific label is denoted $c \in [C] = \{1, \ldots, C\}$. The counts of w and constituent wordpieces s_t in each class c are denoted $n(w, c)$ and $n(s_t, c)$. Based on these counts, we define the following multinomial distributions:

$$p(c|\bullet) = n(\bullet, c) / \textstyle\sum_{c'} n(\bullet, c') \quad (1)$$

where $\bullet = w, s_t$, etc. Based on this we define the entropy

$$H(\bullet) = -\textstyle\sum_c p(c|\bullet) \log p(c|\bullet) \quad (2)$$

Suppose $H(w)$ is small. This means w is potentially a good feature for the LRL task. Now suppose a wordpiece s_t has large $H(s_t)$. That means s_t is being shared across other words that are dis-

tributed more evenly across classes. If this is the case for most s_t, then fragmentation of w may be a serious problem. To combine information from all wordpieces, we average their entropies, and use the *relative increase in entropy*, going from LRL word to wordpieces, as one signal for the danger of fragmenting w. As an example, suppose the word 'धरम' (religion) occurs ten times in a three-class sentiment analysis dataset with the class distribution of 'positive', 'neutral', and 'negative' as (1,1,8). Its wordpieces have class distributions 'ध' (100,85,80), '##र' (130,235,250), and '##म' (130,90,125). Then as per equation 2, $H(\text{'धरम'}) = 0.639$, $H(\text{'ध'}) = 1.094$, $H(\text{'##र'}) = 1.062$, and $H(\text{'##र'}) = 1.086$. The average wordpiece entropy is $H_S(\text{'धरम'}) = \frac{1.094+1.062+1.086}{3} = 1.081$, and the percentage of entropy reduction from average wordpiece to word entropy is about 41%.

We also retain two simpler signals: the number of fragments $|\mathcal{T}(w)|$, and the frequency of w in the LRL task corpus. LRL words are sorted on the amount of entropy decrease and the top LRL words proposed for vocabulary augmentation. We remove words with very low frequency and retain a prefix of specified size to obtain V_{new}, the LRL words to be added to the MLLM vocabulary. Algorithm 1 shows a high-level pseudocode.

3.2 Embedding initialization

Here we describe the different ways to initialize the embeddings of newly-added LRL words.

InitLRL: The embedding of the newly-introduced LRL word is initialized using other LRL wordpieces already in the MLLM dictionary. Suppose we add Bengali word 'হাসপাতাল', ('hospital' in English). Suppose the existing MLLM tokenizer splits it into ['হ', '##াস', '##প', '##াত', '##াল']. Then we initialize the embedding of 'হাসপাতাল' with the average of the existing MLLM embeddings of the fragments.

InitHRL: Here we translate 'হাসপাতাল' to English ('hospital'), tokenize it using $\mathcal{T}$, and take the average embedding of the tokens in the list.

InitMix: We use the average of InitLRL and InitHRL embeddings.

InitRand: We randomly initialize the embeddings of the newly-added words.

It is challenging to learn good contextual embedding for words in V_{new} due to very small task-specific training data compared to the MLLM pre-training corpus. Therefore, we found it necessary to apply some regularization to avoid overfitting during fine-tuning. Let $\mathcal{T}, \mathcal{T}'$ be the initial and final MLLM tokenizers. For a particular sentence $S = w_1, w_2, ..., w_I$ with words w_i, we will get two different tokenizations; these will generally lead to different contextual embeddings $E = (e_1, \ldots, e_K)$ and $E' = (e'_1, \ldots, e'_L)$; generally $K \neq L$. We average-pool these to get vectors $\boldsymbol{e}, \boldsymbol{e}'$ which a final layer uses for the classification task, with losses $\ell_{\mathcal{T}}$ and $\ell_{\mathcal{T}'}$. We also use $(\boldsymbol{e}+\boldsymbol{e}')/2$ for a third classification, with loss ℓ_{mix}. The overall training loss is $\ell_{\mathcal{T}} + \ell_{\mathcal{T}'} + \ell_{\text{mix}}$, where $\ell_{\mathcal{T}}$ and ℓ_{mix} are expected to reduce overfitting.

Task name	LRL(s)	#Train	#Test
(a) IITP product review	Hindi	4182	523
(b) Bengali sentiment	Bengali	12576	1587
(c) Bengali hate speech	Bengali	981	295
(d) Gujarati headlines	Gujarati	5269	659
(e) Malayalam headlines	Malayalam	5036	630
(f) GLUECos sentiment	Hi,En,code-mix	10079	1260

Table 2: Salient statistics of tasks. Note the small size of LRL datasets. Further details in Table 6.

4 Experiments

4.1 Datasets and evaluation metric

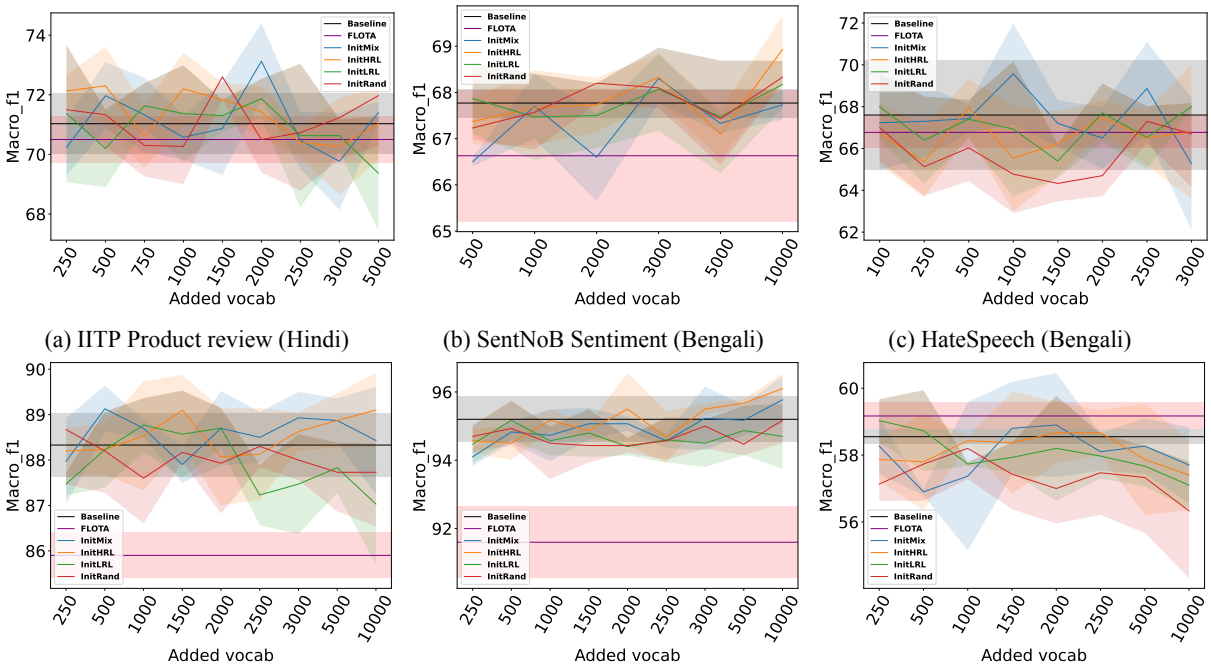
We experiment with six short multi-class text classification tasks covering four Indian languages and a Hindi-English code-mixed dataset. We show the details of the datasets in Tables 2 and 6. We use mBERT as the MLLM and report macro-F1 (we report the accuracy metric in Appendix B). Details of model hyperparameters are present in Appendix C.

4.2 Quantitative results

In Figure 1, we plot macro-F1 against the extent of vocabulary augmentation. Green, orange, and blue lines show the performance with InitLRL, InitHRL, and InitMix initialization, respectively. Corresponding colored bands show 1-standard deviation spreads.

V_{new} helps: For all tasks, including V_{new} is better than baseline MLLM, and the gap is usually significant. This shows that even minimal training of newly added LRL tokens that used to be UNK or over-fragmented helps improve performance.

More augmentation$\not\Rightarrow$larger lift: We expected that larger V_{new} would monotonically improve performance, but this was not universally the case. Inclusion of non-informative words, as we grow V_{new} (Δ_H decreases with high variance as shown in Appendix B Figure 3), maybe a reason.



(a) IITP Product review (Hindi) (b) SentNoB Sentiment (Bengali) (c) HateSpeech (Bengali)

(d) Headline prediction (Gujarati) (e) Headline prediction (Malayalam) (f) GLUECoS (Hindi-English code-mix)

Figure 1: Macro F1 vs. increasing LRL words added to MLLM dictionary. Black = MLLM baseline. Purple = FLOTA. The blue, orange, green, and red line represents the performance with V_{new} with different levels of augmentation and embedding initialization. Solid lines = average. The shaded region shows the standard deviation over three random runs.

Sentence	Added vocab/Δ_H/towards	Gold/Pred$_{\text{EVALM}}$/Pred$_{\text{Baseline}}$
मे दुवा करता हुन की भाई आप का ये muvi सैन का rekod टॉड डे **Translation:** I pray that this movie of yours breaks Sen's record.	'दुवा'/100%/Positive	Positive/Positive/Neutral
উওরটা আমিও পেরেছি , তবে ৮-১০ সেকেন্ড সময় লেগেছে **Translation:** I also get the answer, but it took 8-10 seconds	'উওরটা'/100%/Positive 'পেরেছি'/100%/Positive	Postive/Positive/Neutral
sub गंदा है पर धंधा है ये . **Translation:** Everything is dirty but it is a business.	'गंदा'/58.5%/Negative	Neutral/Negative/Neutral
সালমা নির্বাচনে হারার পর থেকেই পুলিশের বিরুদ্ধে একের পর এক সরযন্ত , হাহাহাহা **Translation:** Ever since Salma lost the election, there has been one after another conspiracy against the police, hahaha	'হাহাহাহা'/100%/Positive 'একের'/100%/Positive	Negative/Positive/Negative

Table 3: Error analysis of EVALM. The first two examples show the win cases of EVALM, where the last two rows are the failure cases compared to baseline predictions. The middle column depicts the added vocab (underlined in the sentence) along with the entropy reduction percentage and the class it mostly belongs to.

Initialization does not matter much: Although there are cases where InitHRL or InitMix performs better than InitLRL, we did not find significant performance difference between different embedding initialization of new LRL words. Transfer of embeddings from a well-represented HRL is the likely reason. We also check the performance by randomly initializing the V_{new} words and find, for almost all the cases, random initialization performance, both for macro-F1(in Figure 1) and accuracy(in Appendix B Figure 2), is lesser compared to InitHRL, InitLRL, or InitMix. It suggests meaningful initialization helps.

Comparison with recent approaches: We compare EVALM with AVocaDo (Hong et al., 2021) keeping V_{new} comparable in size. Table 4 shows that AVocaDo leads to performance *degradation* for all LRL datasets. The lack of domain-specificity for our datasets may be why AVocaDo's performance dropped. We also compare with FLOTA (Hofmann et al., 2022) in Figure 1. For all datasets except GLUECoS Hi-En codemix dataset, EVALM performs better than FLOTA. A possible explanation is that mBERT vocabulary already includes many English as well as Hindi words, which helps FLOTA better compose embeddings of morphological components of English and Hindi words compared to other Indian languages.

Tasks→	(a)	(b)	(c)	(d)	(e)	(f)
$V_{\textbf{new}}$	2000	10000	1000	500	10000	250
EVALM	73.13	68.93	69.57	89.13	96.10	59.03
AVocaDo	68.01	64.99	60.27	85.54	91.81	54.85

Table 4: Here last two rows show the performance between best performing model of EVALM with AVocaDo. (a)–(f) are the datasets/tasks defined in Table 2.

Tasks→	(a)	(b)	(c)	(d)	(e)	(f)
EVALM	73.13	68.93	69.57	89.13	96.10	59.03
$-\ell_{reg}$	71.53	66.30	68.67	88.63	92.23	56.47

Table 5: Ablation. The first and second rows show our best model performance, trained with/without ℓ_{reg} respectively.

Regularization helps: Table 5 shows that EVALM with AVocaDo-style regularization performs better than without it, for all datasets.

Cases where EVALM hurts: The samples in Table 3 show that EVALM generally helps by spotting words important for predicting the correct class. This is shown in the first two examples, where the added vocabulary (Δ_H=100%) tipped the prediction toward the gold label. But the last two examples show cases where for a word, the train and test set frequency distribution among target classes are different. As a consequence, these words may become misleading at test time.

5 Conclusion

We have proposed a simple and effective method to augment an MLLM wordpiece vocabulary with LRL words that are important for LRL classification tasks. Our study, involving several Indian languages, shows a consistent positive impact of vocabulary augmentation and fine-tuning. We find more augmentation does not guarantee performance improvement, and different embedding initialization fails to show significant performance differences among themselves. We also show that regularization is crucial to prevent overfitting new LRL word embeddings during fine-tuning. We have limited the augmentation to whole LRL words, and a judicious selection of LRL wordpieces may improve performance. We also want to extend to other target tasks (especially language generation) and a more diverse set of LRLs.

6 Limitations

While EVALM demonstrates that vocabulary augmentation with LRL task performance as objective requires different priorities from vocabulary augmentation for improving representation for its own sake, our work opens up several avenues for exploration. Our understanding of the potential conflict between fidelity of LRL word representation from wordpieces and LRL task class discrimination requirements remains far from complete, particularly when we extend from sequence-to-single-label applications to sequence labeling (as in POS and NER tagging) and further to sequence-to-sequence applications (such as translation). Perhaps, further experiments with mBERT and other MLLMs will further our understanding of these trade-offs. While initializing an LRL word embedding using InitHRL or InitMix, we depend on automatic machine translation, which can be error-prone. Ranking by Δ_H and picking a prefix fails to discount informative but correlated features. A more sophisticated formulation of loss of information owing to fragmentation, taking multiple LRL words into account simultaneously, may alleviate this problem. In the short term, these two limitations may deserve closer scrutiny.

References

Gaurav Arora. 2020. iNLTK: Natural language toolkit for indic languages. In *Proceedings of Second Workshop for NLP Open Source Software (NLP-OSS)*, pages 66–71, Online. Association for Computational Linguistics.

Ethan C. Chau, Lucy H. Lin, and Noah A. Smith. 2020. Parsing with multilingual BERT, a small corpus, and a small treebank. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1324–1334, Online. Association for Computational Linguistics.

Hyung Won Chung, Dan Garrette, Kiat Chuan Tan, and Jason Riesa. 2020. Improving multilingual models with language-clustered vocabularies. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4536–4546, Online. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Abteen Ebrahimi and Katharina Kann. 2021. How to adapt your pretrained multilingual model to 1600 languages. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*

and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 4555–4567, Online. Association for Computational Linguistics.

Valentin Hofmann, Janet Pierrehumbert, and Hinrich Schütze. 2021. Superbizarre is not superb: Derivational morphology improves BERT's interpretation of complex words. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3594–3608, Online. Association for Computational Linguistics.

Valentin Hofmann, Hinrich Schütze, and Janet Pierrehumbert. 2022. An embarrassingly simple method to mitigate undesirable properties of pretrained language model tokenizers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*.

Jimin Hong, TaeHee Kim, Hyesu Lim, and Jaegul Choo. 2021. AVocaDo: Strategy for adapting vocabulary to downstream domain. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4692–4700, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Khondoker Ittehadul Islam, Md Saiful Islam, Sudipta Kar, and Mohammad Ruhul Amin. 2021. Sentnob: A dataset for analysing sentiment on noisy bangla texts. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, Gokul N.C., Avik Bhattacharyya, Mitesh M. Khapra, and Pratyush Kumar. 2020. IndicNLPSuite: Monolingual Corpora, Evaluation Benchmarks and Pre-trained Multilingual Language Models for Indian Languages. In *Findings of EMNLP*.

Md. Rezaul Karim, Bharathi Raja Chakravarti, John P. McCrae, and Michael Cochez. 2020. Classification benchmarks for under-resourced bengali language based on multichannel convolutional-lstm network. In *7th IEEE International Conference on Data Science and Advanced Analytics (IEEE DSAA,2020)*.

Simran Khanuja, Sandipan Dandapat, Anirudh Srinivasan, Sunayana Sitaram, and Monojit Choudhury. 2020. GLUECoS: An evaluation benchmark for code-switched NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3575–3585, Online. Association for Computational Linguistics.

Xin Liu, Baosong Yang, Dayiheng Liu, Haibo Zhang, Weihua Luo, Min Zhang, Haiying Zhang, and Jinsong Su. 2021. Bridging subword gaps in pretrain-finetune paradigm for natural language generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6001–6011, Online. Association for Computational Linguistics.

Benjamin Minixhofer, Fabian Paischer, and Navid Rekabsaz. 2021. Wechsel: Effective initialization of subword embeddings for cross-lingual transfer of monolingual language models.

Sangwhan Moon and Naoaki Okazaki. 2020. PatchBERT: Just-in-time, out-of-vocabulary patching. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7846–7852, Online. Association for Computational Linguistics.

Elena Sofia Ruzzetti, Leonardo Ranaldi, Michele Mastromattei, Francesca Fallucchi, and Fabio Massimo Zanzotto. 2021. Lacking the embedding of a word? look it up into a traditional dictionary.

Vin Sachidananda, Jason Kessler, and Yi-An Lai. 2021. Efficient domain adaptation of language models via adaptive tokenization. In *Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing*, pages 155–165, Virtual. Association for Computational Linguistics.

Wen Tai, H. T. Kung, Xin Dong, Marcus Comiter, and Chang-Fu Kuo. 2020. exBERT: Extending pre-trained models with domain-specific vocabulary under constrained training resources. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1433–1439, Online. Association for Computational Linguistics.

Hai Wang, Dian Yu, Kai Sun, Jianshu Chen, and Dong Yu. 2019. Improving pre-trained multilingual model with vocabulary expansion. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 316–327, Hong Kong, China. Association for Computational Linguistics.

Zihan Wang, Karthikeyan K, Stephen Mayhew, and Dan Roth. 2020. Extending multilingual BERT to low-resource languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2649–2656, Online. Association for Computational Linguistics.

Wenhao Yu, Chenguang Zhu, Yuwei Fang, Donghan Yu, Shuohang Wang, Yichong Xu, Michael Zeng, and Meng Jiang. 2021. Dict-bert: Enhancing language model pre-training with dictionary.

Entropy-guided Vocabulary Augmentation of Multilingual Language Models for Low-resource Tasks
(Appendix)

A Discussion on vulnerable LRL words

We discuss some natural ideas to determine LRL words vulnerable to improper wordpiece segmentation.
UNK and fragment counts: A natural impulse may be to augment the vocabulary with all LRL words (in the task's train fold) that cannot be assembled from wordpieces in the original vocabulary (i.e., those that become UNK tokens). This is neither necessary nor sufficient. Many UNK words may offer little signal toward labelling. As an example, suppose the Bengali word 'দোয়া' (translate to 'prayer' in English) split to a single ['[UNK]'] token after passing through mBERT tokenizer can be helpful for sentiment analysis classification task. But the word 'জেলায়' (translate to 'in the district' in English) also split to a single ['[UNK]'] token but might not carry any particular signal for the sentiment classification. On the other hand, simply adding all LRL characters as 'wordpieces' precludes UNKs entirely but by no means assures us that the LRL words thus assembled will obtain contextual embeddings of good quality. The word 'ভালবাসা' (translate to 'love' in English) splits to ['ভ', '##াল', '##বা', '##সা'], where all these word fragments do not carry any semantic meaning in Bengali.
Contextual embedding distortion: Another natural idea is to ask if embeddings of wordpieces assembled into the LRL word can be combined by the (typically transformer-like) MLLM network into a good-quality embedding for the LRL word. This can be ascertained only if we have access to a reference embedding for the LRL word, which can be obtained only after introducing the LRL word into the vocabulary and re-training the MLLM! Another problem with this approach is that it is not guided by the impact of the distortion of embedding of a LRL word on end-task accuracy.

	Tasks	Language	Train instances	Test instances
(a)	**IITP Product Review (Kakwani et al., 2020)**	Hindi	4182	523
(b)	**Bengali Sentiment Analysis (Islam et al., 2021)**	Bengali	12576	1587
(c)	**Bengali HateSpeech (Karim et al., 2020)**	Bengali	981	295
(d)	**Gujarati headline classification (Arora, 2020)**	Gujarati	5269	659
(e)	**Malayalam headline classification (Arora, 2020)**	Malayalam	5036	630
(f)	**GLUECoS Sentiment Analysis (Khanuja et al., 2020)**	Hindi-English code-mix	10079	1260

Table 6: Salient statistics of tasks. Note the small size of LRL datasets.

B Supplementary results

In Figure 2, we report the accuracy with vocab augmentation under different embedding initialization techniques for all the datasets.

Tasks→	**(a)**	**(b)**	**(c)**	**(d)**	**(e)**	**(f)**
$V_{\textbf{New}}$	500	3000	1000	1500	1000	1500
EVALM	5.58	5.76	4.40	5.81	8.70	4.26
AVocaDo	3.35	4.07	3.65	3.62	4.08	3.69

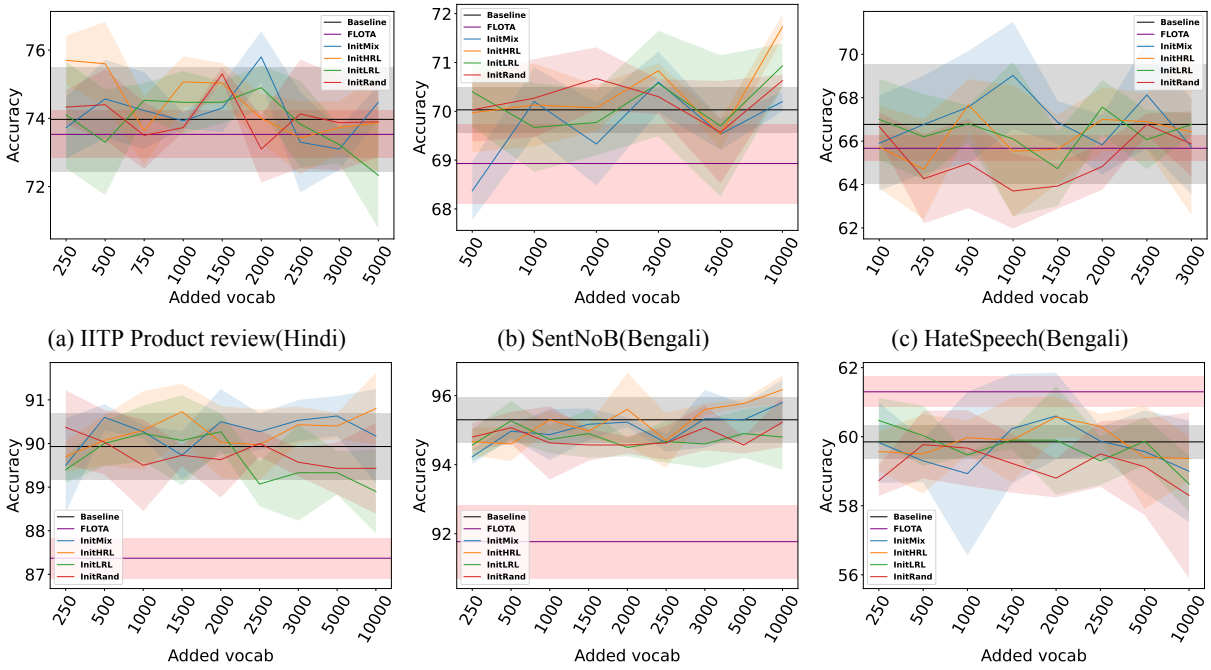
Table 7: Here we compare the average length of the tokens added in our best performing EVALM model with AVocaDo. It shows except one all the cases AVocaDo generates smaller tokens than EVALM. (a)–(f) are the datasets/tasks defined in Table 2.

C Experimental Settings

In all experiments, we trained the models on a single NVIDIA RTX A6000 with 48GB of memory. We implemented all models with PyTorch using the Transformers library from Huggingface. Our model has ∼29M trainable parameters, and it takes 10-45 minutes to train, depending on the size of the datasets.

C.1 Hyperparameters

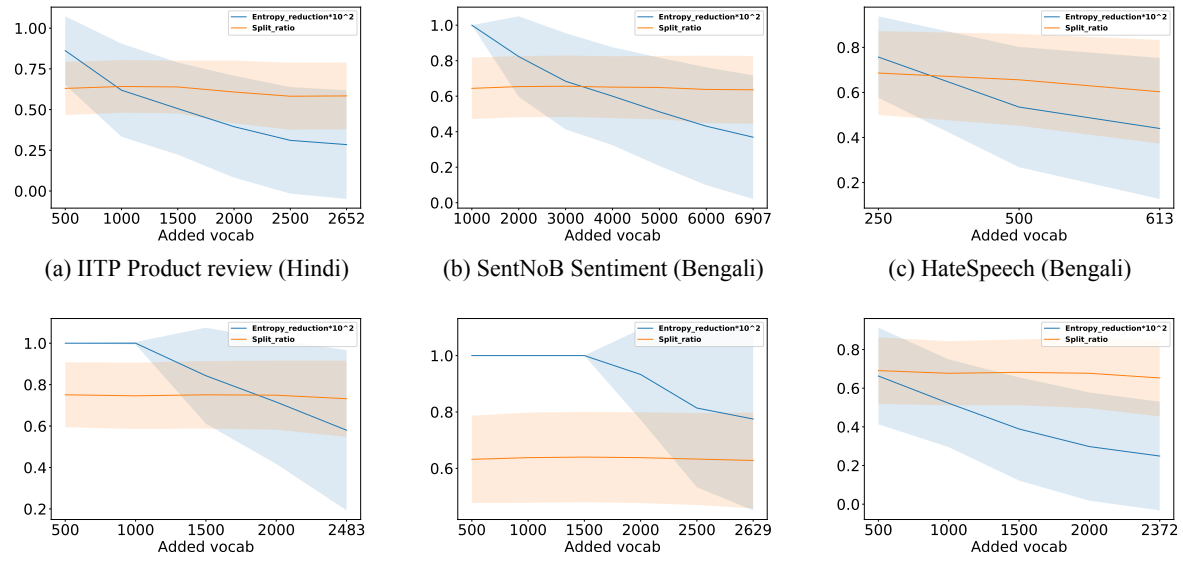
We search for the best hyperparameters manually based on the macro F1 scores. These parameter values are listed in Table 8.



(a) IITP Product review(Hindi) (b) SentNoB(Bengali) (c) HateSpeech(Bengali)

(d) Headline prediction(Gujarati) (e) Headline prediction(Malayalam) (f) GLUECoS Hindi-English code-mix)

Figure 2: Accuracy vs. increasing LRL words added to MLLM dictionary. Black = MLLM baseline. Purple = FLOTA. The blue, orange, green, and red line represents the performance with V_{New} with different levels of augmentation and embedding initialization. Solid lines = average. The shaded region shows the standard deviation over three random runs.



(a) IITP Product review (Hindi) (b) SentNoB Sentiment (Bengali) (c) HateSpeech (Bengali)

(d) Headline prediction (Gujarati) (e) Headline prediction (Malayalam) (f) GLUECoS (Hindi-English code-mix)

Figure 3: Entropy reduction percentage ($\Delta_H(w)$) and Split ratio $\left(\frac{\mathcal{T}(w)}{n(w)}\right)$ plot for the increasing amount of vocabulary addition throughout datasets. The blue and orange line represents the $\Delta_H(w)$ and $\frac{\mathcal{T}(w)}{n(w)}$, respectively. Solid lines = average. The shaded region shows the standard deviation.

Hyperparameter	**Value**
mBERT version	bert-base-multilingual-cased
Batch size	16, 32
Epoch	15
Learning rate	2×10^{-5}, 5×10^{-5}
max_seq_len	128
θ	1
γ	25

Table 8: Hyperparameters used in experiments. We find the best hyperparameter settings using manual search according to macro F1 performance.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section 6 Limitations

☐ A2. Did you discuss any potential risks of your work?
Not applicable. Left blank.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Abstract and section 1 introduction

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

☐ B1. Did you cite the creators of artifacts you used?
No response.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No response.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No response.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
No response.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No response.

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
No response.

C ☑ Did you run computational experiments?

Section 4 experiments

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Appendix C Experimental Settings

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Appendix C Experimental Settings

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Section 4 experiments

☐ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Not applicable. Left blank.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.