4-2023

Morphologically-aware vocabulary reduction of word embeddings

Chong Cher CHIA
Singapore Management University, ccchia.2018@phdis.smu.edu.sg

Maksim TKACHENKO
Singapore Management University, mtkachenko@smu.edu.sg

Hady Wirawan LAUW
Singapore Management University, hadywlauw@smu.edu.sg

Morphologically-Aware Vocabulary Reduction of Word Embeddings

Chong Cher Chia
Singapore Management University
ccchia.2018@smu.edu.sg

Maksim Tkachenko
Singapore Management University
maksim.tkatchenko@gmail.com

Hady W. Lauw
Singapore Management University
hadywlauw@smu.edu.sg

***Abstract*—We propose SubText, a compression mechanism via vocabulary reduction. The crux is to judiciously select a subset of word embeddings which support the reconstruction of the remaining word embeddings based on their form alone. The proposed algorithm considers the preservation of the original embeddings, as well as a word's relationship to other words that are morphologically or semantically similar. Comprehensive evaluation of the compressed vocabulary reveals SubText's efficacy on diverse tasks over traditional vocabulary reduction techniques, as validated on English, as well as a collection of inflected languages.**

***Index Terms*—word embeddings, compression, vocabulary reduction**

I. Introduction

Word embeddings (Word2Vec [1], GloVe [2]) have found widespread adoption in various NLP tasks such as text classification [3], speech recognition [4], semantic relatedness [5], sentiment analysis [6], question answering [7], etc. NLP-based applications which require pretrained word embeddings are moving ever closer to end-users; for instance, voice recognition on smart devices is now commonplace. The emerging trends of edge computing and on-device AI [8] lead to storing and processing data on the device itself. Advantages include greater responsiveness by avoiding network latencies, untethering from the network when desired, and privacy preservation with less data movement. Models stored and run on-device would thus be small(er) due to small device memory sizes.

Our objective in this paper is reducing the storage size of a given set of word embeddings while limiting NLP task performance loss. Moreover, the intent is to do so in a task-agnostic way to increase reusability, as multiple tasks on the same device could then share the word embeddings. Efforts at reducing the size of word embeddings must necessarily target one of the following: the vocabulary size, embedding dimensionality, or the numerical precision of each dimension (see Section II). These factors are orthogonal and can be used in conjunction. In this work, we focus on the relatively less explored strategy of vocabulary reduction, keeping the original dimensionality and precision as-is.

Our proposed framework, SubText[1], retains a small subset of the original vocabulary, thus reducing the memory footprint proportionately. The size can be specified, and we experiment with sizes two orders of magnitude smaller than the full embedding vocabulary. As any other word not retained is reconstructed from the retained subset, the core issue thus rests on a judicious selection of which of subset of the vocabulary ought to be retained.

The rest of the paper is organized as follows. Section III formally defines the problem. Section IV describes a morphologically-informed reconstruction of an out-of-vocabulary word from the retained words which does not feature vocabulary-specific parameters, and does not grow in size with the retained vocabulary. Hence, the memory footprint lies only within the proportion of the words retained. To select the subset of words that optimize the reconstruction objective, in Section V we propose SubText, which preserves morphological and semantic neighborhoods, and outline how to run its computation efficiently. Section VI compares SubText to other word selection approaches, on various tasks such as word reconstruction, word similarity and word classification as well as intent classification. We further experiment on a collection of 12 inflected languages that consistently outperform baselines, signifying a broader applicability of our proposed approach. Lastly, while the focus of the work is primarily on learnt word embeddings, we conduct a preliminary study on the potential applicability of SubText for compressing word pieces in deep learning models such as BERT [9] in Section VII.

As a statement of contributions, this work advances the literature on word embeddings compression by proposing SubText for vocabulary reduction, and showing its efficacy on various tasks involving multiple languages.

II. Related Work

The memory footprint of word embeddings $X \in \mathbb{R}^{n \times d}$ is proportional to vocabulary size n, word vector dimensionality d, and precision of real-valued dimension (assumed 32 bits), i.e., $32dn$ bits. Compressing X necessarily involves a reduction of these factors. We focus on vocabulary reduction, i.e., retaining a subset of $n' \ll n$ word vectors. Below, we survey other orthogonal approaches.

Dimensionality Reduction To reduce d, a common way is to learn low-dimensional basis vectors to reconstruct high-dimensional word vectors. A linear formulation factorizes X into DA, where $A \in \mathbb{R}^{k \times d}$ is a dictionary of k basis vectors and $D \in \mathbb{R}^{n \times k}$ is a code matrix aligning each word to a linear combination of k basis vectors. The reduction in size comes from $(kn + dk) \ll dn$. For further efficiency, one could factor

[1] https://github.com/preferredAI/subtext

in sparsity [10] or discretization [11], [12]. [13] organizes the k basis vectors into m blocks of $\frac{k}{m}$ basis vectors; savings come from fewer effective basis vectors ($m \ll k$) and fewer bits to encode the weight of each ($\log \frac{k}{m} \ll 32$). [14] uses a neural decoder. [15] allows variable-length codes. [16] uses common words as basis vectors. This is not vocabulary reduction per se, as a code matrix still has to be learnt. The compression can be supervised [17], but our concern is unsupervised, with tasks yet unknown.

Precision Reduction This reduces the numerical precision of each word vector dimension. Truncation [18] rounds floating-point numbers to an arbitrary number of bits (4 to 8 bits). Another avenue is quantization, where each word vector dimension is limited to one of k discrete 'steps'. If each dimension has its own independent quantization, the reduction in size is given by $(nd \log k + 32dk) \ll 32nd$. Product quantization [19], [20] generalizes the scheme above, splitting the dimensionality d into s segments. Within a segment, words are clustered by the relevant dimensions. The gain over quantizing individual dimensions comes from there being only $s \ll d$ segments. Yet another angle is iterative quantization [21], [22], which seeks similarity-preserving binary codes.

Others There are other works whose objectives are not compression. To generate embeddings for out-of-vocabulary (OOV) words, FastText [23] expands the vocabulary, violating the compression objective. [24] employs a similar idea, hence retaining a large number of subwords embeddings. [25] does not seek vocabulary reduction. [26], [27] requires separate models to represent OOV words.

WordPiece [28], BPE [29] and SentencePiece [30] generate vocabulary of a user-specified size and could represent OOV words as segments of the learnt pieces. Morfessor [31] finds morphological segmentations of words but do not allow specifying the compression rate. LMVR [32] extends Morfessor FlatCat [33] to give more control over the output lexicon size. These methods can be thought as text tokenizers and require to learn embeddings post-factum, while SubText preserves the original embedding space and allows graceful compression by dropping words from the vocabulary iteratively. Orthogonal to many approaches, SubText can be applied to aforementioned methods to further reduce their memory footprint.

Others use context vectors to improve task performance [7], [34], bring similar words closer [35], adapt to sentiment lexicon [6], [36], sparsify the representation for interpretability [37]. Techniques like inverted index pruning [38], [39] are not compatible with word embeddings.

III. Problem Formulation

As input, we consider pretrained word embeddings (e.g., Word2Vec). $W = \{w_1, \ldots, w_n\}$ denotes the vocabulary, and $x_w \in \mathbb{R}^d$ represents d-dimensional embedding for each $w \in W$. Alternatively, we use index notation to relate embedding x_i to its associated word w_i. Our objective is to select a subset $W' \subset W$ to be used to reconstruct embeddings of the discarded words $W \setminus W'$.

Let $f_{W'}(w)$ be a reconstruction function that outputs an approximation of x_w based on selected subset W'. Our objective, thus, is to minimize any information loss between $f_{W'}(w)$ and original embedding x_w. As word embeddings often serve as building blocks for a variety of NLP applications, we aim to maximize in vivo performance across several tasks and datasets (see Section VI), assuming that the target task is not yet known. To perform in vitro analysis and to estimate the information loss of reconstruction, we can use any standard $\mathbb{R}^d$ measures. In general, the quality of reconstruction by $f_{W'}$ with respect to a specific word w is expressed as some notion of reconstruction loss, denoted as $q(x_w, f_{W'}(w))$. Lower values indicate better reconstruction, with 0 being perfect reconstruction. A reconstruction function itself may incur storage cost by, for example, memorizing word-to-word correspondence or certain embedding parameters. As this may ultimately defy the purpose of compressing, we restrict our search to functions independent of the vocabulary size and negligible in the context of selected embeddings. We discuss such reconstructions in Section IV.

With the overall objective being the reconstruction of W, we can express our problem as finding a subset $W' \subset W$ that minimizes the aggregated loss.

$$\underset{W' \subset W}{\arg\min} \sum\nolimits_{i=1}^{n} q(x_i, f_{W'}(w_i)). \tag{1}$$

In turn, in Section V, we develop more concrete definitions for $q(x_w, f_{W'}(w))$ and offer a word selection algorithm towards the above optimization problem.

IV. Word Reconstruction

For compression, $f_{W'}(\,\cdot\,)$ should be compact. If $|f_{W'}|$ denotes the number of bits required to encode $f_{W'}$, then $|f_{W'}|$ should be negligible in the context of the selected embeddings. We postulate $|f_{W'}| = \mathcal{O}(1)$ and $|f_{W'}| \ll repr(W')$, where $repr(W')$ is the number of bits required to store the word embeddings for W'.

A naïve candidate may ignore word-specific information, returning the embedding of a predefined "unknown" word (i.e., ⟨UNK⟩) for any $w \notin W'$. Formally,

$$f_{W'}(w) = \begin{cases} x_w, & \text{when } w \in W' \\ x_{\langle \texttt{UNK} \rangle}, & \text{otherwise.} \end{cases} \tag{2}$$

Words made up of common morphemes may overlap in meaning, as they may share similar contexts within a corpus. With a string distance (e.g., Levenshtein) function $g(\,\cdot\,,\,\cdot\,)$, one may define a reconstruction function yielding the embedding of an orthographically close word:

$$f_{W'}(w) = x_{w^\star} \quad \text{for} \quad w^\star = \underset{\bar{w} \in W'}{\arg\min}\, g(w, \bar{w}), \tag{3}$$

However, single-word approximations seem limiting for inflected words, e.g., 'countercheck' with a prefix ('counter') modifying its stem ('check'). Though both morphemes are critical to the meaning, the orthographic approach only selects one. This limitation can be overcome by approximating the target with some composition of embeddings. After the target word is broken into known strings $\bar{w}_i \in W'$, $w = \bar{w}_1 \oplus \cdots \oplus \bar{w}_k$

($\oplus$ is the string concatenation operator), the approximation (we focus on linear compositions in this work) is as follows:

$$f_{W'}(w) = \sum\nolimits_{i=1}^{k} \theta_{w\bar{w}_i} x_{\bar{w}_i} \tag{4}$$

for some composition parameters $\{\theta_{w\bar{w}_i}\}_{i=1}^{k}$. Since we are unable to memorize these parameters within the function definition, they must be computable during inference. We observe that longer morphemes usually capture dominant semantics of the word, and tie the composition parameters to its length: $\theta_{w\bar{w}_i} = \frac{|\bar{w}_i|}{|w|}$.

Still, there are multiple ways to break up a target word into pieces, each leading to different approximations. One way to determine the 'best' decomposition would be to minimize Equation 4 by comparing all possible target decompositions to the original word embedding (observed during training):

$$f_{W'}(w) = \underset{\substack{1 \le k \le |w| \\ \bar{w}_1 \oplus \cdots \oplus \bar{w}_k = w}}{\arg\min} \; d\left(x_w, x\right), \text{where } x = \sum_{i=1}^{k} \frac{|\bar{w}_i|}{|w|} x_{\bar{w}_i} \tag{5}$$

The measure $d\left(x_w, x\right)$ indicates how close x is to the original x_w. We employ cosine distance in our experiments, and alternative measures include L_2 distance.

As the main objective is to evaluate the selected vocabulary W', for the purpose of evaluation, Section VI uses such reconstruction unless otherwise stated.

V. Vocabulary Reduction

Optimizing the embedding reconstruction loss in Equation 1 directly by fully evaluating all $\binom{n}{k}$ subsets where $k = |W'|$ is computationally challenging. We therefore employ a greedy approach for approximation, by removing the word incurring minimal loss every iteration until the desired target size is achieved. In order to preferentially remove uncommon pieces at each iteration, we weigh the reconstruction cost for each word w by how frequently it appears within other words in the training vocabulary (γ_w):

$$h(W') = \underset{w \in W'}{\arg\min} \left(\gamma_w \times q\big(x_w, f_{W' \setminus w}(w)\big)\right) \tag{6}$$

Word Selection Next, we define of the per-word reconstruction loss.

Self-Reconstruction An intuitive starting point is for the reconstruction to be close to its original embedding (i.e., $q\left(x_w, f_{W' \setminus w}(w)\right) = d\big(x_w, f_{W' \setminus w}(w)\big)$). While useful, this criterion alone is myopically focused on one word at a time.

Morphological Neighbours The removal of w from W' renders it unavailable as a building block for other words. Given $w_i, w_j \in W'$, we say w_j is a morphological neighbor of w_i if w_i is a substring of w_j or vice versa. Let $C_{W'}(w)$ be the set of all morphological neighbors of w within W'. To remove w from W', we consider how x_w and morphological neighbors $C_{W'}(w)$ will now be reconstructed.

The removal of words with either few or easily reconstructed neighbors is encouraged by updating the selection criterion thus:

$$\begin{aligned} q\big(x_w, f_{W' \setminus w}(w)\big) &= d\big(x_w, f_{W' \setminus w}(w)\big) \\ &\quad + \sum\nolimits_{u \in C_{W'}(w)} d\big(x_u, f_{W' \setminus \{w,u\}}(u)\big). \end{aligned} \tag{7}$$

Semantic Neighbors Words with similar embeddings are likely to be semantically related [1]. For each word w in W, we define its set of positive (N_w^+) and negative (N_w^-) semantic neighbours (as elaborated below). To preserve the "meaning" of w post-reconstruction, we further seek to preserve the closeness between the reconstruction of w and those of its positive semantic neighbors while keeping w apart from its negative neighbours. We obtain an updated word selection criterion below. $\alpha \in [0, 1]$ is a parameter controlling the relative importance of semantic neighbourhood.

$$\begin{aligned} q\big(x_w, f_{W' \setminus w}(w)\big) &= \alpha\Big(\sum_{u \in N_w^+} d(f_{W'}(w), f_{W'}(u)) - \\ &\sum_{v \in N_w^-} d(f_{W'}(w), f_{W'}(v))\Big) + (1-\alpha)\Big(d(x_w, f_{W'}(w)) + \\ &\sum_{u \in C_{W'}(w)} d\big(x_u, f_{W' \setminus \{w,u\}}(u)\big)\Big). \end{aligned} \tag{8}$$

We postulate that semantic neighbours should constitute a small portion of vocabulary W and possibly capture important traits of the target word. Considering words that contain the target w, we subsequently identify the top m closest candidates (as determined by cosine distance) to w to form the positive neighbors, and randomly sample m from the remaining words as negative neighbors. If there are insufficient negative candidates, we pad them randomly.

Optimization Algorithm The greedy approximate optimizer (as described in Section V), while seemingly straightforward, is not tractable for large vocabularies as $f_{W' \setminus w}(\,\cdot\,)$ in Equation 6 has to be recomputed at each pass of the loop. This is due to the active vocabulary changing from W' to $W' \setminus w$, affecting the reconstruction of other words. We employ an inverted index to look up the set of words a word part is involved in. At the earlier stages of optimization, SubText tends to remove longer words with few morphological neighbours, and the optimizer performs only a handful of updates. Later, as SubText begins removing smaller words (with potentially more morphological neighbours), the vocabulary size is considerably smaller and the algorithm remains tractable.

We employ a min-heap data structure to identify the word with minimal reconstruction cost for each round. Thus, updating the word reconstruction cost uses $O(\log(|W'|))$ operations.

Lazy Optimization To realize a further gain in computational efficiency, we develop a lazy optimization scheme. The summation over $C_{W'}(w)$ in Equation 8 iterates over a large subset of W' when w is short and thus a potential reconstruction building block to many other words. Since $|C_{W'}(w)|$ reduces as words are removed from W', we avoid computing the full summation for some words. We rewrite Equation 8 as follows:

$$\begin{aligned} q\big(x_w, f_{W' \setminus w}(w)\big) &= L(w, W') + G(w, W'), \text{ where} \\ G(w, W') &= (1-\alpha) \sum\nolimits_{u \in C_{W'}(w)} d\big(x_u, f_{W' \setminus \{w,u\}}(u)\big), \\ L(w, W') &= (1-\alpha)\, d(x_w, f_{W'}(w)) \\ &\quad + \alpha\Big(\sum\nolimits_{u \in N_w^+} d(x_u, f_{W'}(u)) \\ &\quad - \sum\nolimits_{v \in N_w^-} d(x_v, f_{W'}(v))\Big). \end{aligned}$$

Since $G(w, W')$ is always non-negative, $L(w, W')$ represents a lower bound of the minimized condition: $L(w, W') + G(w, W') \geq L(w, W')$.

Instead of fully calculating Equation 8, we compute its lower bound $L(w, W')$ for every word $w \in W'$ and iteratively update it with the actual minimized condition. Assuming that the words are indexed in ascending order ($L(w_i, W') < L(w_{i+1}, W')$ for any i), we observe that some computations become redundant when the updated bound becomes lower than the next lower bound: $L(w_j, W') + G(w_j, W') < L(w_{j+1}, W')$ (i.e., we have sufficient information to remove w_j or a predecessor ahead of w_{j+1} without computing $G(w_{j+1}, W')$). As such, we only keep track of words (among those with updated bounds) which satisfies the following condition (as well as minimizing Equation 8):

$$w = \underset{w_i:\, i<j}{\arg\min} \left[L(w_i, W') + G(w_i, W') \right]$$

Running Time We focus on the quality of the compressed word embeddings instead of the computational efficiency of the compression process, which can be done offline. Indicatively, on Intel Xeon 2.20GHz CPU and Nvidia Tesla P100 GPU, SubText requires approximately 9.5 hours to compress the training Word2Vec embeddings, as compared to 10 hours for Semantic Clustering. However, Semantic Clustering has to be run once for each compression level. In contrast, SubText stores the word removal sequence after the first run, and can repeat the word removals for any desired level of compression within minutes.

VI. Experiments

Datasets We involve public datasets as outlined below.

Word2Vec We use Word2Vec [1] embeddings trained on the English Wikipedia due to its large size and wide adoption. We use the top 10% most frequent words (from 1,463,808 unique words) to constitute our observed (*training vocabulary*) W, from which we select W'. We also select the 1,000 most frequent words from the remaining 90% to form a dedicated *testing vocabulary*.

Polyglot We also use Polyglot [40] embeddings, which provides 64-dimensional word embeddings trained on Wikipedia corpora of different languages. We focus on inflected/fusional languages [41] that possess at least 50,000 vocabulary words. Table I shows the language code, name and embedding vocabulary size for the 12 selected languages. We use the top 25,000 most frequent words as our *training vocabulary* W, from which we select W', and the 1,000 most frequent words never seen during training as our *testing vocabulary*. We do not present Polyglot English embeddings, as their results substantially mirror those of (English) Word2Vec.

Baselines Vocabulary reduction as a technique is rather novel, we look to traditional vocabulary reduction baselines for an apple-to-apple comparison.

Corpus Frequency memorizes the most frequent word types, as they are most likely to be encountered later.

Morphological Clustering attempts to minimize Equation 1 assuming an orthographic reconstruction function as defined in Equation 3 (Section IV). We use Levenshtein distance and use Voronoi iteration method [42] to induce clustering.

TABLE I: Language code, name and vocabulary sizes for Polyglot embeddings

Code	Language Name (in English)	Vocabulary Size
bn	Bengali	55,004
de	German	100,004
el	Greek, Modern (1453-)	100,004
es	Spanish; Castilian	100,004
fr	French	100,004
hi	Hindi	94,004
is	Icelandic	55,004
it	Italian	100,004
la	Latin	64,004
pt	Portuguese	100,004
ro	Romanian	100,004
sq	Albanian	82,004

Semantic Clustering is a variant of k-medoid selection, where the words are selected via the embeddings space rather than orthographically. We use cosine distance to assess embedding closeness, so as to evenly cover the embedding space and approximate discarded embeddings with the closest selected embedding.

To arrive at the selection $W' \subset W$ of a given size, all baseline algorithms start from the same base of 66 word types, comprising one-character words present in the corpus ('a', 'b', etc.) as well as the ⟨UNK⟩ token. The selection methods would add words to match the target size (10,000 to 25,000 words).

Since we focus on evaluating the selection methodologies, we employ the method-agnostic reconstruction scheme as defined in Section IV.

A. English Language Embeddings

Word Reconstruction We measure the ability of the selected W' to reconstruct word embeddings. Following Equation 1, we compute the *reconstruction error*: average cosine distance over the set of fixed vocabulary W (lower is better). SubText requires parameter setting (Section V), and we set $\alpha = 0.9$ after grid search. $\alpha > 0$ performs well indicating the contribution of semantic neighborhood. Table II shows results of reconstruction on different target sizes for training and testing vocabularies. On both counts, SubText outperforms the baselines with a steady margin. Corpus Frequency outperforms the clustering baselines that only slightly improves upon random.The low reconstruction error of SubText on training vocabulary validates greedy optimization algorithm as an approximate minimizer for Equation 1. The low reconstruction error on the testing vocabulary suggests potential for SubText to handle out-of-vocabulary words.

Word Relatedness To assess if the proximity between semantically-related words in the original embedding space is preserved in the reconstructed embedding space, we compute pairwise distances between testing vocabulary words in both the original and reconstructed spaces. We then evaluate the selection methods according to standard word relatedness measure by the Pearson correlation coefficient r (higher is better).Table III shows that SubText consistently outperforms the baselines, more so on smaller vocabulary sizes. While larger vocabulary sizes generally lead to increasing positive correlations, both clustering methods yield marginal improvements.

TABLE II: Word reconstruction (lower is better)

Size	Training Vocabulary				Testing Vocabulary			
	Corpus Frequency	Morphological Clustering	Semantic Clustering	SubText	Corpus Frequency	Morphological Clustering	Semantic Clustering	SubText
10K	0.837	0.958	0.957	**0.785**	0.855	0.943	0.941	**0.800**
15K	0.815	0.942	0.939	**0.768**	0.835	0.929	0.925	**0.787**
20K	0.802	0.922	0.923	**0.752**	0.820	0.910	0.910	**0.778**
25K	0.792	0.911	0.912	**0.739**	0.809	0.906	0.903	**0.772**

TABLE III: Word relatedness (higher is better)

Size	Corpus Frequency	Morphological Clustering	Semantic Clustering	SubText
10K	0.113	0.029	0.033	**0.160**
15K	0.136	0.030	0.033	**0.159**
20K	0.144	0.036	0.041	**0.167**
25K	0.152	0.047	0.045	**0.173**

When the vocabulary size is large, there are more common words between any two selections, diminishing differences.

Token Classification We introduce a token classification task spanning three different domains: 20News, BioASQ, and arXiv. *20News*[2] contains news documents, and each sample contains 20 classes with 200 train and 300 test words per class. *BioASQ Task 7a* [43] contains biomedical paper abstracts and Medical Subject Headings, which we use as classes. Each sample contains 16 classes with 100 train and 300 test words per class. *arXiv*[3] contains Computer Science paper abstracts and subject classes for each paper, which we use as classes. Each sample contains 6 classes of 300 train and 200 test words per class.

Since we focus on evaluating embedding reconstruction, we use k-nearest neighbors algorithm (k-NN), a non-parametric classification model. As metric, we use mean reciprocal rank (MRR) over word classes with the words being queries. To obtain ranks from k-NN, we estimate class probabilities and sort predictions in descending order. Ties are resolved by awarding the lowest possible rank for predictions of the same probability. We use $k = 100$ for all k-NN experiments.

The baseline column in Table IV shows the average MRR for Corpus Frequency across five data samples, and other columns show percentage performance with respect to the baseline (100% for Corpus Frequency). Due to size constraints, we only show results at 25,000, but observe similar trends up to 10,000. We indicate statistically significant improvement over baseline (via paired t-test over all samples) at $p = 0.05$ with †, and at $p = 0.1$ with ‡. SubText outperforms the baselines across datasets and sizes with statistically significant improvements.

Sentence Completion We introduce a sentence completion task. The task represents a multiple choice question: given a sentence with a skipped word, which provided choice best fills the gap? The number of choices corresponds to the number of classes. For each incorrect class, we uniformly sample a word of that class from the testing vocabulary. We fix the context window size on either side by experimenting on sizes {3, 5, 7}, and select 7 for best performance on the baseline. To mitigate gains attributed to the classification model, we solely utilize embeddings for prediction. We average all word embeddings in the context window (context embedding), and rank choices by the cosine similarity between the context embedding and the reconstructed choice embeddings.Table IV also shows average MRR for the baseline, and relative performance in the remaining columns. The results largely mirror the trends reported on the token classification task, and SubText outperforms the baselines on all the datasets across sizes with an improvement rate of 2-4%. Note that we repeated this experiment with English Polyglot embeddings, and observed similar trends.

Alternative Reconstruction Function So far, we focus on reconstruction-agnostic evaluation. To experiment with another possible approach, we hypothesize that longer word pieces are better at differentiating word meaning, and rank all compositions by the length of the longest word piece, favouring those with the longest piece nearest to the start of the word in case of ties. We then use Equation 4 to compute the top composition's reconstruction. We report results on word classification and sentence completion (Table V), but we observe the same trend on the word reconstruction and relatedness tasks. The target vocabulary size is 25,000. SubText continues to outperform the baseline, indicating that the selected vocabulary is important regardless of the reconstruction method.

Intent Classification To evaluate the compressed embeddings in vivo, we look at a potential application for a smart device. The *CLINC150* dataset [44] contains sentences annotated with 151 intent classes (e.g., play music). We control the proportion of tokens that must be reconstructed. For each sentence, we mask tokens longer than 2 characters with a probability of *P*. To derive an embedding for each masked token, we temporarily remove it from the compressed vocabulary and place it back thereafter. Samples are generated by repeating this masking process 5 times. We use the average of word embeddings as the sentence embedding and k-NN as a classifier. Table VI shows the average MRR. SubText outperforms the baseline, and the margin of improvement increases with *P*.

B. Inflected Language Embeddings

We repeat the above experiments with pretrained Polyglot embeddings, compressed by SubText to 5,000 words due to the smaller training vocabulary sizes.

Word Reconstruction Table VII shows the *reconstruction error* for each language, for both the Polyglot training and testing vocabulary. SubText outperforms the baselines consistently, similar to the experiment using Word2Vec embeddings (Table II). We note that the *reconstruction error* for the

[2] qwone.com/~jason/20Newsgroups/ [3] kaggle.com/neelshah18/arxivdataset

TABLE IV: Evaluation Tasks at 25K (higher is better), relative to baseline (100%)

Dataset	Token Classification				Sentence Completion			
	(Baseline) Corpus Frequency	Morpho-logical Clustering	Semantic Cluster-ing	SubText	(Baseline) Corpus Frequency	Morpho-logical Clustering	Semantic Cluster-ing	SubText
20News	0.242	82.50	82.84	**105.52**†	0.288	86.15	86.04	**102.66**†
BioASQ	0.273	90.13	90.08	**102.91**†	0.263	95.52	95.37	**103.70**†
arXiv	0.451	95.21	95.81	**101.05**†	0.452	96.23	96.09	**102.21**‡

TABLE V: Alternative reconstruction (higher is better), relative to baseline (100%)

Dataset	Word Classification		Sentence Completion	
	(Baseline) Corpus Frequency	SubText(%)	(Baseline) Corpus Frequency	SubText(%)
20News	0.227	**105.36**†	0.268	**101.28**‡
BioASQ	0.258	**101.35**†	0.252	**102.65**†
arXiv	0.447	**102.17**†	0.442	**100.76**

TABLE VI: Intent classification (higher is better), relative to baseline (100%)

P	(Baseline) Corpus Frequency	SubText(%)
0.25	0.567	**100.48**‡
0.50	0.501	**101.82**†
0.75	0.491	**104.73**†

training vocabulary is relatively lower for Polyglot embeddings as compared to Word2Vec embeddings. This is possibly due to Polyglot embeddings having a smaller vocabulary, thus requiring less compression to reach the target size.

Word Relatedness Next, we calculate the word pair similarities from each Polyglot testing vocabulary using original and reconstructed word embeddings, and show the embedding distance correlation in Table VIII. SubText shows higher correlation as compared to the baseline methods, between the original and reconstructed embedding distances for all selected languages. This suggests that SubText is better able to preserve the proximity of semantically-related words in the reconstructed embedding space.

Case Study Next, we analyse the reconstruction choices Corpus Frequency and SubText made for a sample of four languages in Table IX. From the German (de) examples, we observe SubText reconstructing the target word `herrenmannschaften` (men's teams) as a combination of the words `herren` (men's) and `mannschaften` (teams), as opposed to mainly `herrenmannschaft` (men's team) when using Corpus Frequency. While the SubText reconstruction may be slightly further away from the target word, the word pieces `herren` and `mannschaften` can then be used to reconstruct other target words containing either of these wordpieces, as opposed to only those containing `herrenmannschaft`. Using fewer wordpieces to cover a larger range of target words with common morphemes also benefits the ability of SubText to reconstruct rare words, as it can now memorize more uncommon morphemes that occur in these rare words. Note that this relies on sematically related words sharing morphemes, a feature of inflected languages.

From the French examples, we observe `individualisé` (individualized) being reconstructed with the morpheme `individu` (individual) by SubText. In contrast, Corpus Frequency mostly relies on unigrams, with the longest wordpiece being `dual`. This example highlights a failure mode for Corpus Frequency; morphemes that best reconstruct the target may be unpopular, and thus not available for use in reconstruction.

We observe similar cases in the Icelandic and Portuguese examples as well. For example, SubText reconstructs `helgidagur` (holiday) by `helgi` (weekend) and `dagur` (day), which is more informative as compared to the Corpus Frequency reconstruction of `hel.g.i.dagur`. A similar Portuguese example is `referenciação` (referencing), which is reconstructed using the morphemes `referencia` (reference) and `ação` (action) by SubText and Corpus Frequency respectively.

VII. Inquiry into Pretrained Language Models

While our work focuses on learnt word representations, we note the recent popularity of approaches which jointly train the word embeddings and classification layers, such as [45]. In this section, we explore the applicability of SubText on such deep learning models.

We first fine-tune[4] the "bert-base-uncased" base model (from HuggingFace Transformers [46]) on the Intent Classification task described in Section VI-A. We adopt the same training, validation and testing splits as per the original Intent Classification dataset. The "bert-base-uncased" model contains 30,522 WordPiece [47] tokens in embedding layer, from which we remove 994 tokens matching the pattern `"[unused[0-9]*]"`. We adopt all of the remaining 29528 tokens (inclusive of special tokens i.e., `[UNK]`, `[PAD]`) as training vocabulary.

As the WordPiece algorithm already seeks to reduce the corpus to its constituent substrings, we modify the definition of morphological neighbours (Section V) to w_j being a morphological neighbor of w_i if and only if w_i is a substring of w_j. As the actual corpus counts of each WordPiece token are not provided, we estimate (via Zipf's Law)

[4] 200 epochs, batch size of 250

TABLE VII: Polyglot: word reconstruction for inflected languages (lower is better)

Code	Training Vocabulary				Testing Vocabulary			
	Corpus Frequency	Morphological Clustering	Semantic Clustering	SubText	Corpus Frequency	Morphological Clustering	Semantic Clustering	SubText
bn	0.643	0.702	0.709	**0.551**	0.910	0.952	0.954	**0.854**
de	0.630	0.683	0.690	**0.578**	0.840	0.921	0.923	**0.795**
el	0.665	0.680	0.681	**0.606**	0.878	0.909	0.909	**0.835**
es	0.654	0.693	0.700	**0.597**	0.870	0.918	0.924	**0.803**
fr	0.641	0.675	0.681	**0.572**	0.862	0.900	0.897	**0.788**
hi	0.561	0.585	0.586	**0.512**	0.803	0.857	0.860	**0.755**
is	0.630	0.639	0.643	**0.549**	0.968	1.008	1.010	**0.868**
it	0.616	0.665	0.667	**0.561**	0.837	0.901	0.901	**0.769**
la	0.622	0.605	0.606	**0.551**	0.956	0.991	0.993	**0.865**
pt	0.623	0.659	0.663	**0.570**	0.856	0.904	0.906	**0.786**
ro	0.611	0.647	0.650	**0.560**	0.839	0.884	0.886	**0.787**
sq	0.586	0.593	0.597	**0.539**	0.865	0.900	0.907	**0.809**

TABLE VIII: Polyglot: word relatedness for inflected languages (higher is better)

Code	Corpus Frequency	Morphological Clustering	Semantic Clustering	SubText
bn	0.047	0.044	0.043	**0.091**
de	0.042	0.011	0.013	**0.069**
el	0.021	0.018	0.017	**0.034**
es	0.024	0.010	0.006	**0.056**
fr	0.030	0.019	0.017	**0.058**
hi	0.095	0.087	0.093	**0.159**
is	0.084	0.029	0.032	**0.180**
it	0.045	0.035	0.036	**0.093**
la	0.086	0.019	0.023	**0.125**
pt	0.048	0.017	0.019	**0.087**
ro	0.054	0.040	0.041	**0.087**
sq	0.077	0.029	0.028	**0.115**

TABLE IX: Polyglot: ORIGINAL RECONSTRUCTION Examples

Language	Target Word	Corpus Frequency	SubText
German (de)	weltcupspringen	weltcups.p.ringe.n	weltcup.spring.en
	herrenmannschaften	herrenmannschaft.e.n	herren.mannschaften
	aufgezeichneten	auf.g.e.zeichneten	auf.gezeichnet.en
	verfassungs	v.erfassung.s	verfassung.s
	opernhauses	opernhaus.e.s	opern.hauses
	herausbrachte	her.ausbrach.te	heraus.brachte
French (fr)	individualisé	i.n.d.i.v.i.dual.i.s.é	individu.al.i.s.é
	qu'environ	qu'en.v.i.r.o.n	qu.'.environ
	languette	la.n.guet.t.e	langue.t.te
	abandons	a.b.an.dons	abandon.s
	l'équipement	l'équipe.m.en.t	l.'.équipement
	découpés	d.é.coupés	découpé.s
Icelandic (is)	hugverkarétt	hug.verkar.é.t.t	hug.verka.rétt
	helgidagur	hel.g.i.dagur	helgi.dagur
	skipulagður	sk.i.p.u.lagður	skipulag.ð.ur
	meðferðarinnar	me.ð.ferðarinnar	meðferðar.innar
	algengum	al.g.engum	algeng.u.m
	flokkakerfi	f.lokka.kerfi	flokka.ker.f.i
Portuguese (pt)	referenciação	r.e.f.e.r.e.n.c.i.ação	referencia.ç.ã.o
	reavaliar	r.ea.v.aliar	re.avaliar
	desembarques	de.s.embarque.s	desembarque.s
	sustentando	su.s.tentando	sustenta.n.do
	reproduzindo	r.e.produzindo	reproduz.indo
	aculturação	a.cult.ur.ação	a.cultura.ç.ã.o

$\gamma_w = (1/R_w) \,/\, \left(\sum_{n=1}^{N}(1/n)\right)$, where R_w is the rank of w and $N = 3 \times 10^9$ (approximate size of BERT training corpus [45]). We also conduct a grid search for parameter setting using the validation dataset, and set $\alpha = 0.3$.

Next, we select W' from the training vocabulary, and use them to replace the embeddings in the base model Token Embedding layer with reconstructed embeddings (Equation 5) as required. Figure 1 illustrates this (in orange), where the embedding for `Tok 2` (i.e, E'_2) is replaced by combining embeddings from other tokens. We then calculate the evaluation task MRR for the updated model at different sizes, stopping once the model is below 90% of the original performance.

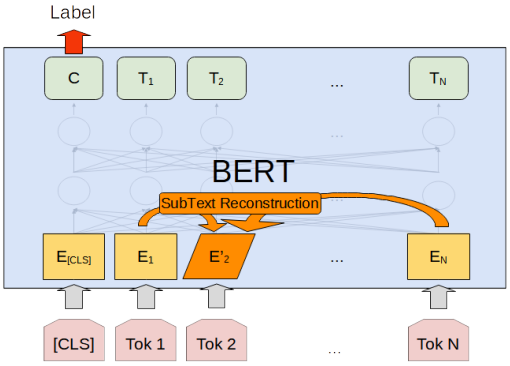

Fig. 1: SubText Reconstruction in Pretrained Language Models

We compare SubText to *Morphological Clustering* and *Semantic Clustering* baselines as previously described in Section VI. However, we are unable to use *Corpus Frequency*, as it would require the corpus used for training "bert-base-uncased". To simulate this, we instead select the top k tokens from the training vocabulary, in the order that they appear in the Word Embedding layer of the base model, with the assumption that these tokens are ordered by frequency (*Frequent Pieces*). The results are reported in Table X.

We first observe that the MRR when using all WordPieces in the baseline model is 0.971. At size 25K (approximately 85% of baseline), both SubText and *Frequent Pieces* perform similarly with an MRR of 0.966. However, the *Morphological Clustering* and *Semantic Clustering* baselines already show significant performance degradation, with a MRR of 0.889 and 0.891 respectively.

For subsequently smaller sizes, we observe that *Frequent Pieces* consistently performs worse than SubText; at 10K tokens (approximately 34% of baseline), *Frequent Pieces* achieved a MRR of 0.879 while SubText achieved 0.905. This suggests that the Word Embedding reconstructions using WordPieces selected by SubText are better able to capture the semantic meaning of the underlying words.

TABLE X: Pretrained Transformer Intent Classification

Size	Frequent Pieces	Morphological Clustering	Semantic Clustering	SubText
10K	0.879	0.321	0.306	**0.905**
15K	0.926	0.558	0.559	**0.943**
20K	0.955	0.762	0.777	**0.959**
25K	**0.966**	0.889	0.891	**0.966**
Full	0.971			

VIII. Conclusion

SubText compresses word embeddings via vocabulary reduction, and preserves original embeddings as well as a word's morphological or semantic neighborhood. Its efficacy is validated on diverse tasks (word reconstruction, word relatedness, word classification, sentence completion and intent classification), demonstrating significant improvements over baselines. Further explorations beyond English into a collection of 12 inflected languages as well as effectiveness in pretrained language models show its potential for broader applicability.

References

[1] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *ICLR Workshop*, 2013.

[2] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *EMNLP*, 2014, pp. 1532–1543.

[3] X. Yang, C. Macdonald, and I. Ounis, "Using word embeddings in twitter election classification," *Information Retrieval Journal*, vol. 21, no. 2-3, pp. 183–207, 2018.

[4] S. Bengio and G. Heigold, "Word embeddings for speech recognition," in *INTERSPEECH*, 2014.

[5] H. T. Nguyen, P. H. Duong, and E. Cambria, "Learning short-text semantic similarity with word embeddings and external knowledge sources," *KBS*, vol. 182, p. 104842, 2019.

[6] M. Tkachenko, C. C. Chia, and H. Lauw, "Searching for the X-factor: Exploring corpus subjectivity for word embeddings," in *ACL*, 2018, pp. 1212–1221.

[7] B. McCann, J. Bradbury, C. Xiong, and R. Socher, "Learned in translation: Contextualized word vectors," in *NeurIPS*, 2017, pp. 6294–6305.

[8] N. I. Mowla, I. Doh, and K. Chae, "On-device ai-based cognitive detection of bio-modality spoofing in medical cyber physical system," *IEEE Access*, vol. 7, 2018.

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[10] D. Yogatama, M. Faruqui, C. Dyer, and N. Smith, "Learning word representations with hierarchical sparse coding," in *ICML*, 2015, pp. 87–96.

[11] J. Tissier, C. Gravier, and A. Habrard, "Near-lossless binarization of word embeddings," in *AAAI*, vol. 33, 2019, pp. 7104–7111.

[12] M. Faruqui, Y. Tsvetkov, D. Yogatama, C. Dyer, and N. A. Smith, "Sparse overcomplete word vector representations," in *ACL-IJCNLP*, 2015, pp. 1491–1500.

[13] R. Shu and H. Nakayama, "Compressing word embeddings via deep compositional code learning," in *ICLR*, 2018.

[14] T. Chen, M. R. Min, and Y. Sun, "Learning k-way d-dimensional discrete codes for compact embedding representations," in *ICML*, 2018, pp. 853–862.

[15] Y. Kim, K.-M. Kim, and S. Lee, "Adaptive compression of word embeddings," in *ACL*, 2020, pp. 3950–3959.

[16] Y. Chen, L. Mou, Y. Xu, G. Li, and Z. Jin, "Compressing neural language models by sparse word representations," in *ACL*, 2016, pp. 226–235.

[17] L. Mou, R. Jia, Y. Xu, G. Li, L. Zhang, and Z. Jin, "Distilling word embeddings: An encoding approach," in *CIKM*, 2016, pp. 1977–1980.

[18] S. Ling, Y. Song, and D. Roth, "Word embeddings with limited memory," in *ACL*, 2016, pp. 387–392.

[19] T. Ge, K. He, Q. Ke, and J. Sun, "Optimized product quantization," *IEEE TPAMI*, vol. 36, no. 4, pp. 744–755, 2013.

[20] K. Shi and K. Yu, "Structured word embedding for low memory neural network language model." in *INTERSPEECH*, 2018, pp. 1254–1258.

[21] S. Liao, J. Chen, Y. Wang, Q. Qiu, and B. Yuan, "Embedding compression with isotropic iterative quantization," in *AAAI*, 2020, pp. 8336–8343.

[22] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin, "Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval," *IEEE TPAMI*, vol. 35, no. 12, pp. 2916–2929, 2012.

[23] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *TACL*, vol. 5, pp. 135–146, 2017.

[24] J. Zhao, S. Mudgal, and Y. Liang, "Generalizing word embeddings using bag of subwords," in *EMNLP*, 2018, pp. 601–606.

[25] T. Luong, R. Socher, and C. Manning, "Better word representations with recursive neural networks for morphology," in *CoNLL*, 2013, pp. 104–113.

[26] Y. Pinter, R. Guthrie, and J. Eisenstein, "Mimicking word embeddings using subword RNNs," in *EMNLP*, 2017, pp. 102–112.

[27] S. Kumar and Y. Tsvetkov, "Von mises-fisher loss for training sequence to sequence models with continuous outputs," in *ICLR*, 2018.

[28] M. Schuster and K. Nakajima, "Japanese and korean voice search," in *ICASSP*. IEEE, 2012, pp. 5149–5152.

[29] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *ACL*, 2016, pp. 1715–1725.

[30] T. Kudo and J. Richardson, "SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," in *EMNLP*, 2018, pp. 66–71.

[31] P. Smit, S. Virpioja, S.-A. Grönroos, and M. Kurimo, "Morfessor 2.0: Toolkit for statistical morphological segmentation," in *EACL*, 2014, pp. 21–24.

[32] D. Ataman and M. Federico, "An evaluation of two vocabulary reduction methods for neural machine translation," in *AMTA*, 2018, pp. 97–110.

[33] S.-A. Grönroos, S. Virpioja, P. Smit, and M. Kurimo, "Morfessor FlatCat: An HMM-based method for unsupervised and semi-supervised learning of morphology," in *COLING*, 2014, pp. 1177–1185.

[34] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *NAACL-HLT*, 2018, pp. 2227–2237.

[35] M. Faruqui, J. Dodge, S. K. Jauhar, C. Dyer, E. Hovy, and N. A. Smith, "Retrofitting word vectors to semantic lexicons," in *NAACL-HLT*, 2015, pp. 1606–1615.

[36] L.-C. Yu, J. Wang, K. R. Lai, and X. Zhang, "Refining word embeddings using intensity scores for sentiment analysis," *TASLP*, vol. 26, no. 3, pp. 671–681, 2017.

[37] F. Sun, J. Guo, Y. Lan, J. Xu, and X. Cheng, "Sparse word embeddings using l1 regularized online learning," in *IJCAI*, 2016, pp. 2915–2921.

[38] D. Carmel, D. Cohen, R. Fagin, E. Farchi, M. Herscovici, Y. S. Maarek, and A. Soffer, "Static index pruning for information retrieval systems," in *SIGIR*, 2001, pp. 43–50.

[39] A. Ntoulas and J. Cho, "Pruning policies for two-tiered inverted index with correctness guarantee," in *SIGIR*, 2007, pp. 191–198.

[40] R. Al-Rfou, B. Perozzi, and S. Skiena, "Polyglot: Distributed word representations for multilingual nlp," in *CoNLL*, 2013, pp. 183–192.

[41] A. Y. Aikhenvald *et al.*, "Typological distinctions in word-formation," *Language typology and syntactic description*, vol. 3, pp. 1–65, 2007.

[42] H.-S. Park and C.-H. Jun, "A simple and fast algorithm for k-medoids clustering," *Expert systems with applications*, vol. 36, no. 2, pp. 3336–3341, 2009.

[43] G. Tsatsaronis, G. Balikas, P. Malakasiotis, I. Partalas, M. Zschunke, M. R. Alvers, D. Weissenborn, A. Krithara, S. Petridis, D. Polychronopoulos *et al.*, "An overview of the bioasq large-scale biomedical semantic indexing and question answering competition," *BMC bioinformatics*, vol. 16, no. 1, p. 138, 2015.

[44] S. Larson, A. Mahendran, J. J. Peper, C. Clarke, A. Lee, P. Hill, J. K. Kummerfeld, K. Leach, M. A. Laurenzano, L. Tang, and J. Mars, "An evaluation dataset for intent classification and out-of-scope prediction," in *EMNLP-IJCNLP*, 2019.

[45] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *NAACL-HLT*, 2019, pp. 4171–4186.

[46] T. Wolf *et al.*, "Transformers: State-of-the-art natural language processing," in *EMNLP*, Oct. 2020, pp. 38–45.

[47] Y. Wu *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016.