Lightweight Adaptation of Neural Language Models via Subspace Embedding

Amit Kumar Jaiswal   Haiming Liu

University of Surrey
University of Southampton

Introduction & Problem Formulation

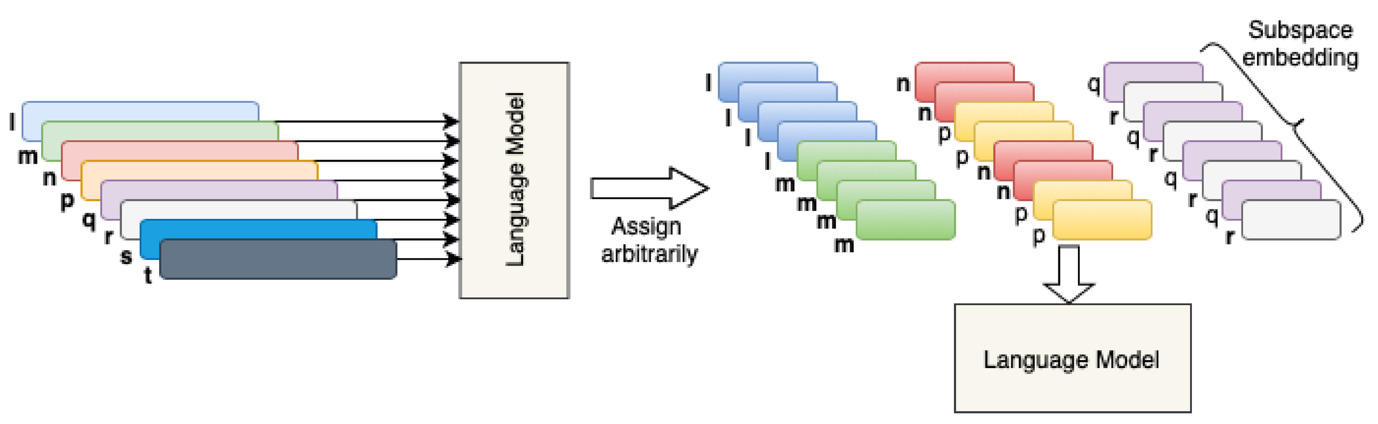
Goal: To reduce the word embedding size in pre-trained language models by representing each word with composition of f low-dimensional embeddings shared between vocabulary.

Spotlight:

- Substantially alleviates the number of embedding parameters in the embedding part through Cartesian product.
- Solves the out-of-vocabulary problem in the (masked) language models.
- Subspace embeddings achieve compression rates beyond 99.8% in comparison with the original embeddings for the language models on XNLI and GLUE benchmark suites.

Problem Settings:

- **Subspace Embedding** (SE) describes the latent space of contextual elements within a token, where each element composes to form the original embedding.
- SE create an arbitrary-sized vector of each word that incorporates semantic relationships.
  - We arbitrarily assign the subspace embedding to each token based on its index and perform a Cartesian product with subspace embedding to construct embedding vectors



- **Calibration of Subspace Embedding:**
  - Original embedding vectors: E_i, E_j, their SE vectors: $\{v_i^f\}, \{v_j^f\}, \forall i, j \in \{1, 2, ..., D\}$
  - Conditions for uniqueness of partitioned embedding vectors: $f \in \{1, 2, ..., F\}$ such that $\{v_i^f\} \neq \{v_j^f\}$ and $i \neq j$
  - A mapping function to transform original embeddings to subspace embeddings, $\mathcal{F} : \mathcal{P} \rightarrow \mathcal{Q} \times \ldots \times \mathcal{Q}$, where a set of the embedding index as $\mathcal{P} \in \{1, 2, ..., D\} \subset \mathrm{N}$ and $\mathcal{Q} = \{1, 2, ..., Q\} \subset \mathrm{N}$ depicts a set of each SE vector index
  - Generalise via Cartesian product, $\mathcal{F}(n) = (c_1 \times c_2 \times \ldots \times c_f)\,\overbrace{(n, \ldots, n)}^{f}$
  - We have f distinct $Q \times (d/f)$ embedding table, where each subspace
  - Subspace embedding representation, $v_n = \oplus_{f=1,\ldots,F} v_{c_f(n)}$, where v_n, v_{c_f} are the corresponding embedding vectors and $\oplus$ denotes the concatenation operation.

Contributions

A word embedding compression method for pre-trained language models (PLMs) that

- allocates shared subspace embedding to the embedding vector in two ways:
  - It allocates sequentially using modulo operation
  - It assigns dispersed subspace embedding using a pre-trained language model with contextual information

Our Approach & Experiments

Techniques for Embedding Compression:

Algorithm 1 Assign Subspace Embedding Arbitrarily

Input: D number of embeddings with dimension d, and set of subspace embeddings F

$Q \leftarrow \lceil D^{1/f} \rceil$ ▷ number of each subspace embedding
Initialise f-th Q subspace embedding vectors $\{v_q^f \in \mathrm{R}^{\frac{d}{f}}\}_{q=1}^{Q}, \forall f \in \{1, \ldots, F\}$
for $n = 1, 2, \ldots, D$ **do**
  for f = 1,2,..., F **do**
    $c_f(n) = (n/Q^{f-1}) \mod Q^f$
  end for
  $v_n = \oplus_{f=1}^{F} v_{c_f(n)}$
end for

Output: The incorporated embedding vectors are $\{v_n\}_{n=1}^{D}$.

Algorithm 2 Cluster-based Subspace Embedding

Input: D number of embeddings, Q number of subspace embeddings, d dimension of embedding, and number of subspace embeddings set F, the pre-trained embedding model $\mathcal{L}_P = \{p_n\}_{n=1}^{D}$

Initialise f-th Q subspace embedding vectors $\{v_q^f \in \mathrm{R}^{\frac{d}{f}}\}_{q=1}^{Q}, \forall f \in \{1, \ldots, F\}$
$c_f(n) \leftarrow 0, \forall f = 1, \ldots, F, n = 1, \ldots, D$
for $f = 1, 2, \ldots, F$ **do**
  extract distinct tuples from $\{\mathcal{F}(n)\}_{n=1}^{D}$
  for distinct $\mathcal{F}(n^*)$ in $\{\mathcal{F}(n)\}_{n=1}^{D}$ **do**
    if $f \neq F$ **then**
      $\{\mathcal{L}_P\}_{\mathcal{F}(n^*)} \leftarrow \{p_n : \mathcal{F}(n) = \mathcal{F}(n^*)\}_{n=1}^{D}$
      alter k-means algorithm to $\{\mathcal{L}_P\}_{\mathcal{F}(n^*)}$
      the outcomes labelling to $c_f(n)$, where $\mathcal{F}(n) = \mathcal{F}(n^*)$
    else
      $c_f(n) \leftarrow$ arbitrary number among Q candidates
    end if
  end for
end for
Collect $v_n = \oplus_{f=1}^{F} v_{c_f(n)}, \forall n \in \mathcal{P}$

Output: The incorporated embedding vectors are $\{v_n\}_{n=1}^{D}$.

Language Model Settings:

Table 1: Description of the altered neural language models.

NLMs	Vocabulary Size	# Embeddings	$\mid \theta \mid$	$\mid \theta_v \mid$
RoBERTa$_S$	50k	50k	51M	25.7M
+2-SE	50k	225	26M	115k
+3-SE	50k	37	26M	18.9k
+8-SE	50k	4	26M	2k
XLM-R$_S$	250k	250k	154M	128M
+3-SE	250k	63	26M	32k

Results on the GLUE Benchmark:

Table 2: Results of Arbitrarily Dispersed Subspace Embedding on GLUE. Columns in blue colour follow Algorithm 1.

Dataset \ Model	RoBERTa$_S$ (Ours)	+2-SE	+3-SE	+4-SE	+6-SE	+8-SE
f	1	2	3	4	6	8
	50k	225	37	15	7	4
SST-2 [21]	89.8	88.4	88.0	88.1	87.2	88.0
Quora Questions[3]	86.5	84.0	83.0	83.3	82.6	83.0
MNLI [28]	79.5	74.3	73.1	72.8	73.5	73.0
QNLI [19]	88.1	84.0	83.4	84.1	84.1	83.0
MRPC [9]	88.3	88.0	85.5	87.4	85.2	86.3
RTE [8]	72.8	66.9	67.8	70.0	67.4	67.8
STS-B [3]	88.0	79.2	77.3	78.4	79.5	76.4
CoLA [26]	38.0	35.6	18.5	23.2	25.5	20.0

Table 3: Results of the Algorithm 2 on GLUE. Shaded columns in red and yellow colour denote the clustered SE using k-means, and uniform cluster size.

Dataset \ Model	RoBERTa$_S$ (Ours)	+2-SE	+3-SE	+3-SE	+3-SE	+3-SE	+3-SE
f	1	2	3	3 Q=100	3 Q=200	3 Q=50	3 Q=100
$\mid \theta_v \mid$	25.7M	115k	18.9k	104k	154k	25.6k	51.2k
% ↓	-	99.5	99.93	99.6	99.3	99.87	99.8
SST-2 [21]	89.8	88.4	88.0	88.2	90.0	89.3	89.3
Quora Questions[4]	86.5	84.0	83.0	84.7	85.6	84.5	84.6
MNLI [28]	79.5	74.3	73.1	75.9	77.5	75.8	77.2
QNLI [19]	88.1	84.0	83.4	85.1	85.5	83.5	85.8
MRPC [9]	88.3	88.0	85.5	87.3	88.6	87.7	87.3
RTE [8]	72.8	66.9	67.8	67.1	69.7	67.9	70.7
STS-B [3]	88.0	79.2	77.3	81.6	84.5	80.1	84.8
CoLA [26]	38.0	35.6	18.5	37.5	34.9	33.6	36.7

Results on Multilingual dataset: We use the XLM-R model based on the Unicoder [3] to evaluate a cross-lingual transfer task. Our altered XLM-R$_S$ network with 250k and 63 number of embeddings for 3-SE, and 128 for clustered SE. The performances on English dataset are 74%, 72.6%, and 72.9% for XLM-R$_S$, 3-SE, and clustered SE.

Conclusion and Future Work:

- Introduced a novel compact embedding structure, which significantly reducing the number of parameters in neural language models.
- We intend to test and scale our embedding compression techniques on LLMs, catering over 200M parameters.
- We evaluated our compact embedding structure on English/Multilingual datasets. Our main structure of the pre-trained language model for downstream tasks follows RoBERTa [1]. Also, we employ XLM-R [2] for performance tests of subspace embedding on multilingual datasets.

References:

[1] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692.

[2] Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., ... & Stoyanov, V. (2020, July). Unsupervised Cross-lingual Representation Learning at Scale. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (pp. 8440-8451).

[3] Huang, H., Liang, Y., Duan, N., Gong, M., Shou, L., Jiang, D., & Zhou, M. (2019, November). Unicoder: A Universal Language Encoder by Pre-training with Multiple Cross-lingual Tasks. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP) (pp. 2485-2494).