Random embeddings baseline for text-level NLP tasks

Lukas Stankevičius[0000–0003–0012–5471] and Mantas Lukoševičius[0000–0001–7963–285X]

Faculty of Informatics, Kaunas University of Technology, Kaunas, Lithuania

Abstract. Transformer models have established themselves as the state-of-the-art deep learning architecture in the field of natural language processing (NLP). Yet, despite huge quantities of data and compute, unsupervised sentence embeddings derived from raw pre-trained states are just little better than random embeddings. In this work, we try to reduce this gap even further by improving sentence vectors derived from random embeddings. In particular, we investigate the effects of different tokenization and vector sizes for semantic textual similarity, short text clustering, and classification tasks. The proposed methods show substantial improvements for classification due to the size of embeddings and also the effect on the tokenizer choice. Our work reinforces the random vectors method as a good and simple baseline. It also shows that for sentence vectors, the performance gap between pre-trained transformer features and simple random vectors is even smaller than previously thought.

Keywords: BERT · embeddings · natural language processing · text embeddings · sentence vector representation · semantic similarity · transformer models · unsupervised learning

1 Introduction

Since the dawn of the transformer model [4, 23], there is a strange contrariety between the performance in downstream tasks depending on whether the model was fine-tuned on the task or not. These huge deep learning models range from 108M [4] trainable parameters and ingest billions of tokens during the pre-training stage. Thus, one would expect that after the training, the loss profile flattens out, the inner states of such a model would eventually become the representation of the input text. If the text is "understood" or, in other words, fine-grained contextualized embedding was acquired, it should result in good performance in both the pre-training task and better results in later downstream ones.

It is difficult to extract the compact representation from the transformer model. There is no single hidden code layer like in autoencoders or the last hidden state as in recurrent neural networks. Transformer models lack the bottleneck property, which could actually be the source of their power. One has to consider L layers, N tokens, and d size vectors in 3-dimensional representation space

$\mathbb{R}^{(L+2)\times N\times d}$. For the early BERT [4] model with $12 + 2$ layers (we also add static token embeddings and input embeddings as layers) and 768 element vectors a single token (sentence has many tokens!) would have a representation of size $12\times 768 = 10\,752$. Not only does one have to know which aggregation to perform, but the basic aggregation of such sequences is also computationally expensive. As a result, current unsupervised methods utilizing inner transformer weights for sentence vectorization tasks lag behind the supervised methods.

In a recent work [19] we proposed a simple random vector baseline with various post-processing techniques, which is very close in performance to unsupervised sentence representations. We showed that the techniques improve both BERT unsupervised representations and random vectors to almost the same level.

In this work, our aim is to improve the random vectors method further by scaling vector sizes and changing tokenization.

Our contributions are:

- We experimentally test multiple token vector sizes for random vectors method.
- We experimentally test the different tokenization effect on random vectors method.
- We propose recommendations for using random vectors as the baseline and share the code for reproducibility.

The remainder of the paper is organized as follows. First, we provide a concise review of the related work in Section 2. Then in Section 3, we outline the experimental setting and give a detailed background on our chosen approach, tokenization, and datasets. In Section 4, we present the results. Finally, we discuss the findings of this work in Section 5.

2 Related Work

The first big leap to deep learning models as we know them today was the Doc2Vec model based on shallow neural networks [11] in 2014. During training, each paragraph was assigned a unique vector, which, together with the context words, was used to predict the target word. The resulting paragraph vectors were compact, unlike the earlier bag-of-words ones.

The next significant shift was to the recurrent neural network (RNN) encoders. Here, RNN would consume text word by word, updating its hidden state each time and producing the final one, which should bear the representation of the entire sequence. Popular models of this type are SkipThought [8] and InferSen [3]. This architecture allowed for the more complex input interactions but was limited as a result of information fading out in long sequences.

Fixes to recurrent neural models included bidirectional setting [16] and an attention mechanism [1], which later evolved into a separate transformer architecture [23] with specific supervised fine-tuning schemes such as SBERT [15] to learn sentence vectors.

There was only a partial success in unsupervised extraction of a representative text embedding from the raw pre-trained transformer model. Sentence-level task improvements were observed by employing whitening [6, 20], applying learned embedding space transformation [13], using spectral filters to dissect representations at different temporal scales [21], and prompting [7, 18]. However, these approaches resulted in limited improvements, compared to models that were directly fine-tuned for sentence-level tasks. Moreover, post-processing techniques have even been criticized for overfitting to specific task types [24]. As a result, unsupervised sentence-level representations are lagging behind supervised ones.

The current top models for sentence-level tasks are supervised and fine-tuned by contrastive learning [5, 12]. The authors of [15] were the first to use Natural Language Inference (NLI) supervision to fine-tune BERT [4]. Almost all later works used NLI datasets, such as [2, 26]. Currently, the main driving force of these models is training on huge amounts of synthesized NLI-type data (positive and negative pairs of data).

The main idea of our work is similar to the work of [25]. The authors of that paper investigated random initializations in LSTM, echo state networks, and bag of random embedding projections, which is the same method we use in this work. They found that these random weight methods score just less than 2% points on average than standard sentence encoders of that time on evaluated tasks.

Unlike [25], we perform a more comprehensive evaluation, incorporating STS and short text clustering tasks. In addition, we experimented with different tokenizers, which come from transformer models and were not available at the time of [25].

Our work is also similar to the previous [19] in that we use the same random vectors model. Yet we test this method further: we experiment with different tokenization and scaling options to make it even more efficient.

3 Methods

Here we present our model, tokenization, and the downstream tasks on which we evalute the resulting representations.

3.1 Random vectors model

This model is very simple and can be summarized in the following steps.

1. Given the token vocabulary size $|V|$ and the token vector dimension d a matrix with the shape $\mathbb{R}^{|V|\times d}$ is randomly initialized from the normal (Gaussian) distribution, centered at 0 with 0.1 standard deviation.
2. For each sequence of token indexes (tokenized text), the corresponding vectors are selected from the previously created matrix, and the entire text representation is then computed as the average vector.

The random vectors model is very fast. This allows changing the parameters $|V|$ (also the corresponding tokenization) and d based on the target text type and domain. We experimented with vector sizes of 3, 48, 192, 3 072 and 12 288 elements. The larger embeddings filled the computer memory for clustering and classification tasks and slowed the whole experiment pipeline while showing only negligible performance differences.

3.2 Tokenization

For the transformer model to work, its vocabulary must be of a manageable size. The bigger the dictionary, the more corresponding vectors the model has to contain and train. BERT [4] model used the tokenization of WordPiece [27], T5 [14] and mT5 [29] - SentencePiece [10] to encode text as WordPiece tokens [9,17], while Llama used SentencePiece with byte pair encoding (BPE) algorithm [17]. All these methods create a fixed-length vocabulary from the most frequent subwords and at the same time minimizing the number of tokens in each sample.

The multilingual model mT5 [29] tries to cover multiple languages and has the largest vocabulary of 250 100 in this work. Its successor ByT5 [28], on the other hand, has the smallest one, as it avoids tokenization at all and instead feeds byte sequences directly into the model.

We also tested the classical tokenization of splitting text into words by spaces. We implemented it by lower-cased the text and then splitting it by white spaces. In Python it is simply achieved using `str.lower().split()`, therefore, we name this tokenization as "lowersplit".

The demonstration of tokenizer properties and example tokenization are shown in Table 1. It shows that although some tokenizers are of comparable size, like T5 and Llama, their produced token pieces are different because of the different corpus and parameters used to train it.

Table 1. Tokenizers used and their properties. $||V||$ is vocabulary size. Example sentence from STS12 task is "a woman mixes up vegetables ."

Tokenizer	$\|\|V\|\|$	Tokenization example
BERT [4]	30 522	[a] [woman] [mixes] [up] [vegetables] [.]
T5 [14]	32 100	[_] [a] [_woman] [_mixes] [_up] [_vegetables] [_] [.]
mT5 [29]	250 100	[_] [a] [_woman] [_mix] [es] [_up] [_] [vegetable] [s] [_] [.]
ByT5 [28]	384	a woman mixes up vegetables .
Llama [22]	32 000	[_a] [_woman] [_mix] [es] [_up] [_veget] [ables] [_.]
lowersplit	∞	[a] [woman] [mixes] [up] [vegetables] [.]

3.3 Evaluation tasks

We tested our sentence vectorization method on eight Semantic Textual Similarity (STS), six short text clustering, and twelve classification tasks. The average

Table 2. Average number of tokens in each sample for different tasks due to the different tokenization.

Dataset	Tokenizer					
	BERT	T5	mT5	ByT5	Llama	lowersplit
STS tasks						
STS12	14	18	19	65	16	12
STS13	11	14	15	54	13	10
STS14	11	15	16	54	14	10
STS15	12	16	17	58	14	11
STS16	14	18	19	65	15	13
STR	16	19	19	69	18	12
Clustering tasks						
agnews	26	39	38	164	36	23
biomedical	20	23	22	90	23	13
googleTS	33	48	45	198	44	28
searchsnippets	24	30	31	144	30	18
stackoverflow	12	13	13	50	12	8
tweet	11	15	14	58	14	9
Classification tasks						
CR	22	27	29	96	23	20
MPQA	3	4	5	19	4	3
MR	26	32	34	115	29	22
MRPC	25	30	33	119	30	22
SCICITE	40	50	49	197	45	31
SICK-E/R	10	12	13	46	11	10
SST2	12	14	15	54	13	10
SST5	23	29	31	103	26	19
STS-B	13	17	18	59	15	11
SUBJ	28	36	38	129	32	25
TREC	11	13	14	49	12	10

number of tokens for each tokenizer and task is depicted in Table 2. In general, all tasks are composed of 3 to 31 words per sample on average. We followed the evaluation setup from [19], please refer to that paper for more details about the tasks.

4 Results

Increasing random vector size improves all tasks considered. For STS and clustering it saturates at about the size of 768, but does not overfit with bigger sizes. For classifaction tasks, improvements were observed up to the last experimented size of 12 288. Only the classification results of 75.9 ± 0.2 are comparable to the sentence vectors sourced from unsupervised BERT weights with a classification accuracy of 79.9 with the difference of just 4%.

Table 3. Effect of tokenization and random vector size for 10 random seed initializations. Both correlation and accuracy scores range form 0 to 100 (from the worst to the best). The best result for each vector size is bolded, while underlined result is the best across all sizes.

Tokenization	Vector size					
	3	48	192	768	3 072	12 288
Avg. Spearman correlation of STS tasks.						
BERT	23.9±1.7	46.4±1.3	49.6±0.7	50.7±0.5	50.8±0.3	50.9±0.1
ByT5	23.9±2.7	44.9±0.8	46.9±0.5	47.1±0.3	47.3±0.1	47.3±0.1
Llama	**26.5±2.0**	**50.9±0.6**	**54.7±0.7**	**55.5±0.4**	**55.7±0.2**	<u>**55.8±0.1**</u>
lowersplit	22.3±2.8	45.5±1.2	47.0±0.7	48.3±0.4	48.4±0.2	48.5±0.1
mT5	21.9±2.6	42.1±2.3	43.2±1.2	43.9±0.4	43.8±0.3	43.9±0.1
T5	23.0±2.8	43.3±2.0	45.4±0.8	46.3±0.5	46.3±0.2	46.4±0.1
Transformer model results from [19]						
Only pre-trained BERT			61.3			
NLI supervised SimCSE			81.5			
Avg. accuracy of clustering tasks.						
BERT	13.6±0.4	**31.7±0.5**	**35.7±0.3**	36.1±0.2	**36.3±0.4**	<u>**36.4±0.2**</u>
ByT5	**14.1±0.3**	19.8±0.4	21.2±0.3	21.6±0.3	21.8±0.1	21.8±0.1
Llama	13.7±0.2	31.1±0.4	35.0±0.5	**36.3±0.3**	36.1±0.2	36.0±0.3
lowersplit	13.2±0.3	28.5±0.3	32.6±0.3	32.8±0.2	33.0±0.4	33.1±0.3
mT5	13.5±0.3	26.5±0.8	28.7±0.5	29.2±0.2	29.4±0.2	29.4±0.2
T5	14.0±0.2	29.1±0.6	31.9±0.5	32.5±0.3	32.7±0.2	32.6±0.2
Transformer model results from [19]						
Only pre-trained BERT			57.0			
NLI supervised SimCSE			59.8			
Avg. accuracy of classification tasks.						
BERT	47.5±0.6	56.7±0.5	**63.7±0.2**	69.5±0.3	74.0±0.1	75.7±0.1
ByT5	47.7±0.5	53.9±0.2	57.3±0.2	59.9±0.2	60.9±0.1	61.5±0.1
Llama	**48.2±0.6**	56.6±0.4	63.5±0.2	**69.8±0.2**	**74.1±0.2**	<u>**75.9±0.2**</u>
lowersplit	47.6±0.3	**57.0±0.2**	63.7±0.4	69.6±0.3	73.6±0.2	75.4±0.2
mT5	47.4±0.2	56.1±0.4	62.8±0.4	69.3±0.2	73.9±0.2	75.8±0.2
T5	47.4±0.4	56.0±0.4	62.9±0.4	69.0±0.2	73.6±0.2	75.5±0.2
Transformer model results from [19]						
Only pre-trained BERT			79.9			
NLI supervised SimCSE			86.5			

An interesting result is that Llama tokenization is better than others for semantic textual similarity tasks. From 48 to 12 288 random vector size, its average Spearman correlation is at least 4.9% higher. Meanwhile, for classification and short text clustering we do not observe any differences.

5 Discussions

In the previous work [19] we reported that random embedding model is comparable to the unsupervised-feature-based BERT approach. We showed that the gap for STS tasks scores is greatly reduced by using various post-processing and token aggregation techniques. In this work, we show that the same is true for classification tasks with the help of larger random vector sizes. This reduced the classification accuracy gap of 10% to just 4%.

We are unable to improve short text clustering tasks. Both vector size scaling and different tokenizers brought only marginal improvements. In [19] there was also little success in using special techniques to boost these vectors for short text clustering, unlike STS. We think that the clustering task may be sensitive to the weaknesses of random vectors: disregard of word order, undiscriminating homonyms, polysemy, and synonyms.

One of the promising aspects of the random vectors model is that, unlike transformers, they are not bound to a particular tokenization. It is possible to learn token splitting rules on the fly, customized to the target task at hand. In this work, we found that this is indeed the case for the STS task, which prefers Llama tokenizer. We think that the reason behind this is the more grammar-aware tokenization of Llama, capturing stems more often (see Table 1) and thus contributing to a better sentence matching.

In our initial experiments, we tried to train a tokenizer for each task separately, using only texts from that particular task. The best vocabulary sizes of such tokenizers were just about several hundreds of tokens (and depended on the task), but we were unable to improve the tokenizers from the table 1. As is the case with Llama and STS tasks in our experiments, we think that besides the domain, tokenizers also need the out-of-the-domain texts that can better lead tokenization to be aware of the grammar. This could be one of the possible future directions of this work.

To use as a baseline for classification tasks, we recommend random vector sizes of at least 3 072 elements. For semantic textual similarity tasks, we recommend sticking with 768 size vectors, but using Llama tokenizer. For both clustering and STS we also advise applying post-processing and representation-shaping techniques, as described in [19]. We hope that this work will help to utilize random vectors in constrained and compute-scarce environments and for tasks that are not sensitive to bag-of-words setting deficiencies.

We make our code available at `https://github.com/LukasStankevicius/Random-embeddings-baseline-for-text-level-NLP-tasks`.

References

1. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: Bengio, Y., LeCun, Y. (eds.) 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015), http://arxiv.org/abs/1409.0473

2. Bowman, S.R., Angeli, G., Potts, C., Manning, C.D.: A large annotated corpus for learning natural language inference. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp. 632–642. Association for Computational Linguistics, Lisbon, Portugal (Sep 2015). https://doi.org/10.18653/v1/D15-1075, https://aclanthology.org/D15-1075
3. Conneau, A., Kiela, D., Schwenk, H., Barrault, L., Bordes, A.: Supervised learning of universal sentence representations from natural language inference data. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. pp. 670–680. Association for Computational Linguistics, Copenhagen, Denmark (Sep 2017). https://doi.org/10.18653/v1/D17-1070, https://www.aclweb.org/anthology/D17-1070
4. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). pp. 4171–4186. Association for Computational Linguistics, Minneapolis, Minnesota (Jun 2019). https://doi.org/10.18653/v1/N19-1423, https://aclanthology.org/N19-1423
5. Gao, T., Yao, X., Chen, D.: SimCSE: Simple contrastive learning of sentence embeddings. In: Empirical Methods in Natural Language Processing (EMNLP) (2021)
6. Huang, J., Tang, D., Zhong, W., Lu, S., Shou, L., Gong, M., Jiang, D., Duan, N.: WhiteningBERT: An easy unsupervised sentence embedding approach. In: Findings of the Association for Computational Linguistics: EMNLP 2021. pp. 238–244. Association for Computational Linguistics, Punta Cana, Dominican Republic (Nov 2021). https://doi.org/10.18653/v1/2021.findings-emnlp.23, https://aclanthology.org/2021.findings-emnlp.23
7. Jiang, T., Jiao, J., Huang, S., Zhang, Z., Wang, D., Zhuang, F., Wei, F., Huang, H., Deng, D., Zhang, Q.: PromptBERT: Improving BERT sentence embeddings with prompts. In: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing. pp. 8826–8837. Association for Computational Linguistics, Abu Dhabi, United Arab Emirates (Dec 2022), https://aclanthology.org/2022.emnlp-main.603
8. Kiros, R., Zhu, Y., Salakhutdinov, R.R., Zemel, R., Urtasun, R., Torralba, A., Fidler, S.: Skip-thought vectors. In: Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 28. Curran Associates, Inc. (2015), https://proceedings.neurips.cc/paper/2015/file/f442d33fa06832082290ad8544a8da27-Paper.pdf
9. Kudo, T.: Subword regularization: Improving neural network translation models with multiple subword candidates. In: Gurevych, I., Miyao, Y. (eds.) Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 66–75. Association for Computational Linguistics, Melbourne, Australia (Jul 2018). https://doi.org/10.18653/v1/P18-1007, https://aclanthology.org/P18-1007/
10. Kudo, T., Richardson, J.: SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In: Blanco, E., Lu, W. (eds.) Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations. pp. 66–71. Association for Computational Linguistics, Brussels, Belgium (Nov 2018). https://doi.org/10.18653/v1/D18-2012, https://aclanthology.org/D18-2012/

11. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: Xing, E.P., Jebara, T. (eds.) Proceedings of the 31st International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 32, pp. 1188–1196. PMLR, Bejing, China (22–24 Jun 2014), https://proceedings.mlr.press/v32/le14.html
12. Lee, C., Roy, R., Xu, M., Raiman, J., Shoeybi, M., Catanzaro, B., Ping, W.: NV-embed: Improved techniques for training LLMs as generalist embedding models. In: The Thirteenth International Conference on Learning Representations (2025), https://openreview.net/forum?id=lgsyLSsDRe
13. Li, B., Zhou, H., He, J., Wang, M., Yang, Y., Li, L.: On the sentence embeddings from pre-trained language models. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 9119–9130. Association for Computational Linguistics, Online (Nov 2020). https://doi.org/10.18653/v1/2020.emnlp-main.733, https://aclanthology.org/2020.emnlp-main.733
14. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J.: Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of Machine Learning Research **21**(140), 1–67 (2020), http://jmlr.org/papers/v21/20-074.html
15. Reimers, N., Gurevych, I.: Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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. 3982–3992. Association for Computational Linguistics, Hong Kong, China (Nov 2019). https://doi.org/10.18653/v1/D19-1410, https://aclanthology.org/D19-1410
16. Schuster, M., Paliwal, K.: Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing **45**(11), 2673–2681 (1997). https://doi.org/10.1109/78.650093
17. Sennrich, R., Haddow, B., Birch, A.: Neural machine translation of rare words with subword units. In: Erk, K., Smith, N.A. (eds.) Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 1715–1725. Association for Computational Linguistics, Berlin, Germany (Aug 2016). https://doi.org/10.18653/v1/P16-1162, https://aclanthology.org/P16-1162/
18. Springer, J.M., Kotha, S., Fried, D., Neubig, G., Raghunathan, A.: Repetition improves language model embeddings. In: The Thirteenth International Conference on Learning Representations (2025), https://openreview.net/forum?id=Ahlrf2HGJR
19. Stankevičius, L., Lukoševičius, M.: Extracting sentence embeddings from pretrained transformer models. Applied Sciences **14**(19) (2024). https://doi.org/10.3390/app14198887, https://www.mdpi.com/2076-3417/14/19/8887
20. Su, J., Cao, J., Liu, W., Ou, Y.: Whitening sentence representations for better semantics and faster retrieval (2021). https://doi.org/10.48550/ARXIV.2103.15316, https://arxiv.org/abs/2103.15316
21. Tamkin, A., Jurafsky, D., Goodman, N.: Language through a prism: A spectral approach for multiscale language representations. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (eds.) Advances in Neural Information Processing Systems. vol. 33, pp. 5492–5504. Curran

Associates, Inc. (2020), https://proceedings.neurips.cc/paper/2020/file/3acb2a202ae4bea8840224e6fce16fd0-Paper.pdf

22. Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., Lample, G.: Llama: Open and efficient foundation language models (2023), https://arxiv.org/abs/2302.13971
23. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.u., Polosukhin, I.: Attention is all you need. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 30. Curran Associates, Inc. (2017), https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf
24. Wang, B., Kuo, C.C.J., Li, H.: Just rank: Rethinking evaluation with word and sentence similarities. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 6060–6077. Association for Computational Linguistics, Dublin, Ireland (May 2022). https://doi.org/10.18653/v1/2022.acl-long.419, https://aclanthology.org/2022.acl-long.419
25. Wieting, J., Kiela, D.: No training required: Exploring random encoders for sentence classification. In: International Conference on Learning Representations (2019), https://openreview.net/forum?id=BkgPajAcY7
26. Williams, A., Nangia, N., Bowman, S.: A broad-coverage challenge corpus for sentence understanding through inference. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). pp. 1112–1122. Association for Computational Linguistics, New Orleans, Louisiana (Jun 2018). https://doi.org/10.18653/v1/N18-1101, https://aclanthology.org/N18-1101
27. Wu, Y., Schuster, M., Chen, Z., Le, Q.V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Łukasz Kaiser, Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G., Hughes, M., Dean, J.: Google's neural machine translation system: Bridging the gap between human and machine translation (2016), https://arxiv.org/abs/1609.08144
28. Xue, L., Barua, A., Constant, N., Al-Rfou, R., Narang, S., Kale, M., Roberts, A., Raffel, C.: ByT5: Towards a token-free future with pre-trained byte-to-byte models. Transactions of the Association for Computational Linguistics **10**, 291–306 (2022). https://doi.org/10.1162/tacl_a_00461, https://aclanthology.org/2022.tacl-1.17/
29. Xue, L., Constant, N., Roberts, A., Kale, M., Al-Rfou, R., Siddhant, A., Barua, A., Raffel, C.: mT5: A massively multilingual pre-trained text-to-text transformer. In: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 483–498. Association for Computational Linguistics, Online (Jun 2021). https://doi.org/10.18653/v1/2021.naacl-main.41, https://aclanthology.org/2021.naacl-main.41