INTERSPEECH 2021
30 August – 3 September, 2021, Brno, Czechia

Information Retrieval for ZeroSpeech 2021: The Submission by University of Wroclaw

Jan Chorowski[1,2], *Grzegorz Ciesielski*[1], *Jarosław Dzikowski*[1], *Adrian Łańcucki*[3], *Ricard Marxer*[4], *Mateusz Opala*[1], *Piotr Pusz*[1], *Paweł Rychlikowski*[1] *and Michał Stypułkowski*[1]

[1]University of Wroclaw, Poland
[2]NavAlgo, France
[3]NVIDIA, Poland
[4]Université de Toulon, Aix Marseille Univ, CNRS, LIS, France

jan.chorowski@cs.uni.wroc.pl, pawel.rychlikowski@cs.uni.wroc.pl

Abstract

We present a number of low-resource approaches to the tasks of the Zero Resource Speech Challenge 2021. We build on the unsupervised representations of speech proposed by the organizers as a baseline, derived from CPC and clustered with the k-means algorithm. We demonstrate that simple methods of refining those representations can narrow the gap, or even improve upon the solutions which use a high computational budget. The results lead to the conclusion that the CPC-derived representations are still too noisy for training language models, but stable enough for simpler forms of pattern matching and retrieval.

Index Terms: ZeroSpeech Challenge, unsupervised learning, information retrieval, spoken language modeling

1. Introduction

The Zero Resource Speech Challenge series (ZeroSpeech) [1, 2] is an initiative with the ultimate goal of building from scratch a system that learns an end-to-end spoken dialogue system for an unknown language, using only sensory information mainly in the form of recordings, and does not use any linguistic resources or knowledge.

The high-level objective of the competition is to learn various qualities of a natural language at different levels of granularity directly from raw audio without any supervision. ZeroSpeech 2021 evaluates speech understanding using the following tasks and datasets:

1. Phonetic ABX (Libri-Light dataset [3]), where the system has to judge whether two phonemes are identical
2. Lexical (sWUGGY dataset) - classifying whether a spoken utterance is a real or misspelled word
3. Semantic (sSIMI dataset) - assessing semantic similarity between two spoken words
4. Syntactic (sBLIMP dataset) - assigning a *grammatical score* to an utterance in such a way that erroneous sentences have lower scores than correct ones

For all tasks it is assumed that spoken corpora (either LibriSpeech [4], or Libri-Light [3]) are the only sources of language knowledge.

The organizers have provided a baseline solution [2], which we adapt and modify in our submission. Raw audio is fed to a Contrastive Predicting Coding (CPC) model [5]. By trying to predict future representations, the CPC model learns useful intermediate representations of speech. These come in the form of an embedding vector emitted every 10 ms. Collected from a large training corpus, the embeddings are then clustered with the k-means algorithm into 50 pseudo-phones. With this pipeline, any unseen audio can be transformed into a stream of pseudo-phones, on which language models may be trained for downstream tasks.

In this paper we present our submission which tries to address all four tasks. We extend the baseline solution in several directions: we refine the intermediate representations, extracted with CPC, to directly improve the ABX scores. We show that such representations can be used to perform simple fuzzy lookups in a large dataset, and even extract some common patterns that serve as pseudo-words. Our approach to the semantic word similarity task is also based on pseudo-words. Instead of pooling the hidden states of the language model, we opt for a direct discovery of pseudo-words in the corpus. These can be embedded with a word2vec-like approach [6] to form a semantic vector for the entire word. Lastly, for the syntax modeling task we use a simple LSTM model similar to the baseline one.

We provide complete source code of our submission at `https://github.com/chorowski-lab/zs2021`.

2. Phonetic Task: Libri-Light ABX

In the ABX task two speech categories A and B are determined by two recordings (e.g., "bit" vs "bet"), and a third recording X has to be correctly assigned to either one of them.

The baseline representations for the ABX task are CPC-derived embedding vectors. In order to improve upon those representations, we introduce two approaches described in the following section.

2.1. Improvements to CPC Representations

Factoring Out Speaker Identities The embeddings produced by CPC contain information about both the phonetic content and speaker identity. In case of ABX, which is a phoneme recognition metric, the latter is irrelevant. We therefore project the embeddings of the baseline model (CPC-big [2]) into the nullspace of a linear speaker classification model to render the embeddings less speaker-sensitive. We perform speaker classification on baseline CPC embeddings with a projection factorized into matrices A and B, where $A \in \mathbb{R}^{D_{inb} \times D_{emb}}$, $B \in \mathbb{R}^{D_{spk} \times D_{inb}}$, D_{emb} is the dimensionality of embeddings and D_{inb} is the linear bottleneck dimensionality. In order to compute ABX, we multiply the CPC-derived embeddings by $A' \in \mathbb{R}^{(D_{emb} - D_{inb}) \times D_{emb}}$, the nullspace matrix of A.

Averaging with Centroids Higher-level tasks of the competition rely on pseudo-phones found by clustering these vectors

http://dx.doi.org/10.21437/Interspeech.2021-1465

Table 1: *ABX error rates (%, cosine distance) for multiple sizes of nullspaces of speaker classification models. The nullspace dimension complements the bottleneck dimension used to train the speaker recognizer.*

Evaluation			Nullspace dimensionality					
			None	**464**	**448**	**416**	**320**	**256**
dev	clean	within	3.38	3.28	**3.25**	3.29	3.26	3.31
	clean	across	4.17	3.98	**3.94**	3.92	3.98	3.99
	other	within	4.81	4.63	**4.60**	4.61	4.62	4.67
	other	across	7.53	7.34	**7.24**	7.24	7.21	7.26

with k-means. Doing so proves useful, so we incorporate some of the outcomes of the clustering back into the dense CPC-derived vectors.

Specifically, we take a weighted average of every dense CPC-derived embedding e in the embedding space with its cluster centroid c_e:

$$\hat{e} = \alpha\, c_e + (1 - \alpha)\, e. \tag{1}$$

This averaging moves every dense embedding towards its assigned centroid proportionally to the distance from it. This aims to include information about the global characteristics of the embedding space coming from clustering without substantial loss of local information, and might be regarded as a simple form of denoising. It does not change the assignment to the closest centroid.

2.2. ABX Experiments

We evaluate both aforementioned improvements on the ZeroSpeech ABX task. To begin with, we extract 512-dimensional embeddings from the second layer of the CPC's autoregressive module. We run each classification experiment for 10 epochs on the *train-clean-100* subset of LibriSpeech.

ZeroSpeech 2021 dev/test sets are subsets of LibriSpeech dev/test sets, respectively. However, for best results (and for replication of the baseline) we had to first compute the embeddings on the full LibriSpeech test set, to allow the model to keep latent state between consecutive utterances. After all embeddigs were computed, we have kept only the ones needed for ZeroSpeech ABX evaluations.

We investigate the variation in ABX scores with respect to the dimensionality of the resulting nullspace, by testing with different bottleneck sizes D_{inb} of the speaker classifier. We achieve the best ABX result when the nullspace size is 448.

Next, we evaluate the influence of averaging with centroids on the ABX scores (Table 3). It also noticeably improves ABX results, and we achieve the highest error reduction when it is combined with a 448-dimensional nullspace projection. We have experimented with both Euclidean and cosine distance metrics when performing k-means and later choosing the closest centroid. Both yield similar results, and we select the centroid according to the cosine distance in subsequent experiments.

Lastly, we evaluate the influence of both methods on supervised speaker and phoneme classification accuracies. In the nullspace approach, both of them are low (Table 2), and increase with the size of the nullspace ($D_{emb} - D_{inb}$). This indicates that after we attempt to make the representations speaker-insensitive, there is still some stray of speaker-related information present in the remaining dimensions. As seen in Table 4, averaging with centroids also reduces phoneme classification accuracy, proportionally to how much the embeddings are altered, both with and without the nullspace projection.

Table 2: *Phoneme and speaker classification accuracies (%) of models after applying factorized projection heads (top) and nullspace matrices of the aforementioned speaker classification models (bottom) on the baseline embeddings (c.f. Table 1).*

Setup	Proj. head bottleneck / Nullspace dimensionality					
	None / 512	**48 / 464**	**64 / 448**	**96 / 416**	**192 / 320**	**256 / 256**
Speakers (fact. proj. head)	84.85	91.17	91.32	91.59	92.02	92.02
Speakers + null space	84.85	29.42	27.91	24.95	19.50	15.56
Phonemes + null space	78.61	78.46	78.36	78.18	77.46	76.86

Table 3: *ABX error rates (%, cosine distance) for weighted averaging of CPC embeddings with centroids. The bottom half shows results combined with the best 448-dimensional nullspace setup. The nullspace dimension is equal to the difference of dimensions between the embeddings and the bottleneck used to train the speaker classifier. Phoneme classification results in Table 4*

Evaluation			Centroid weight					
			None	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**
dev	clean	within	3.43	3.23	3.16	**3.09**	3.07	3.22
	clean	across	4.20	3.96	3.84	**3.76**	3.77	3.97
	other	within	4.84	4.64	4.62	**4.61**	4.75	5.19
	other	across	7.63	7.38	7.28	**7.32**	7.53	7.93
dev	clean	within	3.25	3.03	2.97	**2.94**	2.93	3.10
+ nullspace	clean	across	3.94	3.66	3.60	**3.58**	3.57	3.75
	other	within	4.60	4.49	4.47	**4.47**	4.66	4.98
	other	across	7.24	7.05	6.94	**7.02**	7.21	7.67

Thus, both tested methods improve ABX, and degrade phoneme separation results. This can be because difficulties of those tasks and power of downstream models used differs - we discard less important parts of the information, which improves ABX results as the task is simple and there are no trained parameters, just representation distances (in which case removing less important parts of information helps) but degrades phoneme separation performance (as we still discard some information which classifier with trainable weights could use). In both cases the relative gain in ABX is bigger than the relative loss in phoneme separation performance.

Table 4: *Phoneme classification accuracies (%) for averaging with centroids, both without the nullspace and after projection to the nullspace. ABX results in Table 3*

Euclidean k-means	Centroid weight					
	None	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**
Phonemes	**78.0**	77.6	77.3	76.9	76.3	75.6
Phonemes + 448-d nullspace	77.7	77.4	77.0	76.6	76.0	75.4

3. Quantization for Higher-level Tasks

The baseline methods for the remaining tasks of higher linguistic levels require quantized inputs, that act as discrete input tokens for language models. This is achieved by clustering CPC-derived vectors with k-means, and mapping every dense vector to its centroid. To achieve the best results on lexical and syntactic tasks (sWUGGY and sBLIMP datasets), we use the CPC-nullspace embeddings instead of the raw CPC embeddings. In contrast to feature extraction for the ABX task, now the LSTM

context in CPC is not kept between the files, as the datasets for specific tasks are not related to one another. Additionally, when computing the distances, we normalize L_2-norm lengths of vectors and in effect switch from the Euclidean metric to the cosine metric for quantization.

For the semantic task (sSIMI dataset), we use the baseline quantizations produced with the raw CPC embeddings, and k-means clustering with the Euclidean metric.

4. Lexical Task: sWUGGY

The goal of the task is to distinguish a real word from a similar pseudo-word in a given pair. Pseudo-words were generated with Wuggy [7], and adhere to orthographic and phonological patterns of the English language. Such pairs make up the sWUGGY dataset, which has two parts: one with real words which appear in the LibriSpeech training set (*base*), and another one in which they do not (*OOV*).

The baseline solution takes pseudo-phones as input, and judges the likelihood of a word with a language model, following [8]. For the *base* pairs, our method performs a dictionary lookup of the quantized representations trying to spot the words in the entire LibriSpeech training corpus.

For the *OOV* pairs, we were trying to use a simple LSTM language model and to combine it with dictionary lookup. But since guessing whether a word is in vocabulary is somehow problematic, and LSTM yielded similar results to DTW, we decided to treat all words in the same way.

4.1. Dictionary Lookup

We build a corpus by pre-processing all LibriSpeech training set utterances to strings of pseudo-phones. For every query word, our goal is to find the closest matching subsequence in the corpus. The lookup is based on dynamic time warping (DTW) [9], and uses subsequence DTW which matches a given sequence to a contiguous subsequence of another, such that the matching cost is minimal across all subsequences. This can be done without any increase in complexity, and is easy to parallelize.

Query words and pseudo-phone representations of training utterances are strings of discrete centroid numbers. A simple similarity matrix between the elements of two sequences x, y would be a binary one. We take advantage of having Euclidean coordinates of the centroids, and compare two pseudo-phones by the similarity of their centroid vectors. Thus, every similarity matrix has real-valued entries, and we perform soft matching of sequences.

We estimate pseudo-log probability of a query word as the negative quotient of the minimum DTW cost to the mean DTW cost of this word. We normalize with the mean DTW cost because longer words tend to have higher costs, which would result in a bias towards shorter words.

4.2. Experiments for sWUGGY

We have tried different lookup methods, such as direct comparison of subsequences or measuring edit distances. Out of the tested methods, the best results were obtained with dynamic time warping. We have also tried to post-process the quantizations, but all attempts worsened the results. This is probably due to loss of useful information, so we run DTW on vanilla quantizations.

For the sWUGGY test set, it was not possible to differentiate between *base* and *OOV*, so we have used only DTW. For *OOV* words, the difference between the results obtained with DTW and LSTM was minor.

Table 5: *Scores for DTW lookup on different quantizations, and with linear and optimal distance matrices. The results were computed for the base dev set of the lexical task (no OOV subset). We used the train-full-960 subset of the LibriSpeech as dictionary.*

Quantization	Distance matrix	Classification accuracy no norm.	norm.
Baseline	none (constant)	68.47%	69.33%
Baseline	Euclidean	68.94%	70.98%
Baseline	Euclidean^2	71.00%	71.64%
Cosine	cosine	72.61%	73.36%
Cosine	cosine$^{1.6}$	73.12%	73.92%

A significant improvement to our DTW relies on a technical detail. If we match the word correctly, we expect the match cost to be spread evenly over the entire sequence. However, when we match the word incorrectly, we expect the cost to be high in some places and low in the others. Thus, we increase the cost of distant pseudo-phones, and decrease the cost of similar ones by raising the cost in distance matrix to some power, which sharpens the distances. In our case, 1.6 was the best for the cosine metric quantization and 2.0 for the baseline quantization.

Processing of the train set took 18 hours on conventional hardware for both baseline and nullspace quantizations. Results presented in Table 5 show that both normalization and modification of the distance matrix yielded a significant improvement in the score.

5. Semantic Task: sSIMI

The goal of the task is to judge the semantic similarity of pairs of words (see [10], [11]). That similarity is then compared with semantic judgments made by the human annotators, collected from a set of 13 existing semantic similarity and relatedness tests (including WordSimilarity-353[12], and mturk-771[13] which was used as a development set).

The submission format encouraged solutions which assign a vector at every temporal location, with a simple pooling method to aggregate them into a vector for the entire recording. The pooling methods included *min*, *max*, *avg*, *last*, etc. In our submission, we have computed a vector for an entire recording, and replicated it along the time axis, so that after pooling with aforementioned functions it would remain unchanged.

Preparation of the Corpus We rely on the baseline pseudo-phone units, extracted with CPC and quantized with k-means. Streams of recognized pseudo-phones contain symbol repetitions, and we treat such blocks as higher order units. We further simplify these sequences by heuristically collapsing subsequent occurrences of the same pseudo-phone, and unify blocks which occur in similar contexts. The treat the result of this procedure as an approximation of a phoneme-level transcription with no word boundaries.

Segmentation We apply segmentation into pseudo-words with SentencePiece [14], which maximizes the probability under a unigram language model [15]. Given a vocabulary $\mathcal{V} = w_1, \ldots, w_n$ with associated occurrence probabilities $(p_1, \ldots, p_n)$ and an utterance x, the most probable segmenta-

tion is determined with the Viterbi algorithm. The vocabulary $\mathcal{V}$ is refined iteratively by maximizing the probability of every utterance under a unigram language model: $P(x) = \prod_i p(x_i)$.

We apply SentencePiece with target vocabulary size 50k. By using ground-truth transcriptions, we found that the corpus has 18,705,420 words, which translates to 1.95 pseudo-word for every real word.

Embedding and Retrieval With the corpus segmented into pseudo-words, we train an ordinary word2vec model [6]. Every recording in the similarity task dataset comprises a single word, and we convert each of them to a sequence of pseudo-phones. Some of those sequences exist in our pseudo-word vocabulary, and already have a unique word embedding calculated with word2vec. Others need to be built from smaller pseudo-word units. A simple way of doing so would be to split the sequence with SentencePiece into known pseudo-words. Knowing that the pseudo-representations tend to be noisy, we instead find the closest matches of each of them in the training corpus wrt. edit distance. Then, word2vec embeddings of these matched pseudo-words are averaged to a single embedding for every input recording.

5.1. Experiments for sSIMI

Since our approach differs from the ZeroSpeech baseline one, we decided to present other word-oriented toplines for sSIMI, that better suit our approach. We compare word vectors trained on a large corpus, LibriSpeech transcriptions, and on our tokenization of LibriSpeech transcriptions. In that last case, we delete spaces and tokenize into 50k units using SentencePiece. The results are presented in Table 6. The embeddings trained on word corpora outperform the RoBERTa topline, which suggests that the proposed approach might deserve further investigation.

Table 6: *Toplines for word-based sSIMI (dev set). First three methods used all word pairs from the dev set, the result for RoBERTa[16] are for the synthetized part of the data.*

Method	**synth.**
Google News word2vec	65.5
LibriSpeech 960 transcriptions (w2v)	36.3
Tokenized Librispeech	16.8
RoBERTa (ZeroSpeech2021 Topline)	32.28

Table 7 presents our results in the ZeroSpeech contest together with other submissions. Our approach achieves the best score in the LibriSpeech subcategory, where the recordings are cut from LibriSpeech and not synthesized. This might indicate that our method is able to discover semantic shades of the known words from the corpus, but unable to generalize further.

6. Syntactic Task: sBLIMP

BLIMP [17] is a challenge dataset for evaluating to what extent language models can understand the English grammar. The dataset consists of pairs of similar sentences, and in every pair only one sentence is correct. In sBLIMP all sentences are synthesized. Since the aim of BLIMP was to evaluate how sentence likelihood is related to its grammatical correctness, there is a natural strategy of solving sBLIMP: use a language model to compute sentence probability and pick the most likely sentence from the pair as the correct one. To this end we use a

Table 7: *Correlation between human judgments and system responses (× 100). For other contestants the best submission on the test part of the data is presented.*

	synth.		**libri.**	
Method	**dev**	**test**	**dev**	**test**
LSTM Baseline	4.42	7.35	7.07	2.38
BERT Baseline	6.25	5.17	4.35	2.48
Ours	5.90	2.42	10.20	9.02
van Niekerk et al.	4.29	9.23	7.69	-1.14
Liu et al.	3.16	7.30	1.79	-4.33
Maekaku et al.	-2.10	6.74	8.89	2.03

LSTM language model trained on quantized nullspace features from LibriSpeech dev subset. In the competition, it had 53% accuracy both on dev and test sets, slightly outperforming the baseline, and being close to 54% of the best submission.

However, this result is not impressive at all: LSTM with random weights has 52.9% accuracy. We hypothesize that this is caused by unbalanced utterance lengths in sBLIMP. We have discovered that incorrect sentences are typically longer than the correct ones. Nevertheless, it is worth saying that BLIMP even in text version is definitely a non-trivial task, as for instance a big LSTM trained on large text corpus achieves only 70% accuracy [18], and even large Transformer[19] model like GPT-2 don't exceed 82% [17, 20].

7. Conclusions and Future Works

We have presented a low-resource, information-retrieval based approach to the tasks of the Zero Resource Speech Challenge 2021. We were able to outperform baselines on every task, and achieve best or close to the best results on all four tasks. Still, many issues deserve further investigation.

First, we can explore the relationship between the ability of neural networks to memorize words, and contrast that with fuzzy information retrieval system. Is it possible to discover the dictionary from recordings, using some combinations of these approaches?

Moreover, we believe that there is a potential synergy with computing semantic vector representation for pseudo-words using word2vec – mainly because it is inexpensive to compute, and moving to bigger datasets can lead to a substantial improvement of embedding quality.

Solving BLIMP in the Zero Resource regime is undoubtedly an ambitious task. We believe that it is worth to consider its simpler, artificially created variants. For instance, a variant in which the incorrect sentences were created by changing word order, or by replacing a randomly chosen word with another. Such simpler task can produce less noisy results.

Lastly, we can explore approach similar to averaging with centroids, but applied during training of CPC. For example by adding a loss based on the distance to simultaneously computed centroids, hoping that its denoising effect will improve the extracted representations.

8. Acknowledgments

The authors thank Polish National Science Center for funding under the OPUS-18 2019/35/B/ST6/04379 grant and the PlGrid consortium for computational resources.

9. References

[1] M. Versteegh, R. Thiolliere, T. Schatz, X. N. Cao, X. Anguera, A. Jansen, and E. Dupoux, "The zero resource speech challenge 2015," in *Interspeech*, 2015.

[2] T. A. Nguyen, M. de Seyssel, P. Rozé, M. Rivière, E. Kharitonov, A. Baevski, E. Dunbar, and E. Dupoux, "The Zero Resource Speech Benchmark 2021: Metrics and baselines for unsupervised spoken language modeling," in *Self-Supervised Learning for Speech and Audio Processing Workshop @ NeurIPS*, 2020.

[3] J. Kahn, M. Rivière, W. Zheng, E. Kharitonov, Q. Xu, P.-E. Mazaré, J. Karadayi, V. Liptchinsky, R. Collobert, C. Fuegen *et al.*, "Libri-light: A benchmark for asr with limited or no supervision," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7669–7673.

[4] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An asr corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 5206–5210.

[5] A. van den Oord, Y. Li, and O. Vinyals, "Representation Learning with Contrastive Predictive Coding," *arXiv:1807.03748 [cs, stat]*, Jul. 2018.

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

[7] E. Keuleers and M. Brysbaert, "Wuggy: A multilingual pseudoword generator," *Behavior research methods*, vol. 42, no. 3, pp. 627–633, 2010.

[8] G. Le Godais, T. Linzen, and E. Dupoux, "Comparing character-level neural language models using a lexical decision task," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. Valencia, Spain: Association for Computational Linguistics, Apr. 2017, pp. 125–130. [Online]. Available: https://www.aclweb.org/anthology/E17-2020

[9] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, no. 1, pp. 43–49, 1978.

[10] M. Baroni, G. Dinu, and G. Kruszewski, "Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Baltimore, Maryland: Association for Computational Linguistics, Jun. 2014, pp. 238–247. [Online]. Available: https://www.aclweb.org/anthology/P14-1023

[11] T. Schnabel, I. Labutov, D. Mimno, and T. Joachims, "Evaluation methods for unsupervised word embeddings," in *Proceedings of the 2015 conference on empirical methods in natural language processing*, 2015, pp. 298–307.

[12] L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, and E. Ruppin, "Placing search in context: the concept revisited." *ACM Trans. Inf. Syst.*, vol. 20, no. 1, pp. 116–131, 2002.

[13] G. Halawi, G. Dror, E. Gabrilovich, and Y. Koren, "Large-scale learning of word relatedness with constraints." in *KDD*, Q. Yang, D. Agarwal, and J. Pei, Eds. ACM, 2012, pp. 1406–1414. [Online]. Available: http://dblp.uni-trier.de/db/conf/kdd/kdd2012.html#HalawiDGK12

[14] T. Kudo and J. Richardson, "SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 66–71. [Online]. Available: https://www.aclweb.org/anthology/D18-2012

[15] T. Kudo, "Subword regularization: Improving neural network translation models with multiple subword candidates," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 66–75. [Online]. Available: https://www.aclweb.org/anthology/P18-1007

[16] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," 2019, cite arxiv:1907.11692. [Online]. Available: http://arxiv.org/abs/1907.11692

[17] A. Warstadt, A. Parrish, H. Liu, A. Mohananey, W. Peng, S.-F. Wang, and S. R. Bowman, "Blimp: The benchmark of linguistic minimal pairs for english," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 377–392, 2020.

[18] K. Gulordava, P. Bojanowski, E. Grave, T. Linzen, and M. Baroni, "Colorless green recurrent networks dream hierarchically," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 1195–1205. [Online]. Available: https://www.aclweb.org/anthology/N18-1108

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.

[20] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," 2019.