Can Pretrained Language Models Understand without Using Prior World Knowledge?

Project Category: Natural Language
Project Mentor: Christopher Wolff

Name: Jeffrey Shen
SUNet ID: jshen2
Department of Computer Science
Stanford University
`jshen2@stanford.edu`

1 Key Information to include

- External collaborators or mentors (if you have any): N/A
- Sharing project with another class: N/A

2 Introduction

Recent studies have seen tremendous success applying large, pretrained language models [1, 2, 3, 4, 5] to various natural language processing (NLP) tasks, achieving superhuman performance on benchmarks such as SuperGLUE [6]. However, these benchmarks often use examples from publicly available text about the real world, meaning that it is possible for models to perform better by understanding language better or by knowing more about the world. In fact, some models, such as T5, are even able to perform closed-book question answering [7], where models are able to answer questions about the world without being provided any context. In this project, we aim to develop a method to randomize entities in task examples so that models cannot rely solely on prior world knowledge.

We focus on BoolQ [8], a text dataset with examples consisting of a question, a passage, and a yes/no answer. Our algorithm takes the dataset and outputs a dataset with perturbed examples, which we call AltBoolQ. We use term frequency and word embeddings to select groups of salient words (e.g. entities) within an example, cluster terms across the dataset using Gaussian mixture models [9], and finally use the clusters to resample terms to obtain perturbed questions and passages. We find surprisingly models trained on BoolQ do not rely solely on world knowledge, and are able to generalize relatively well to AltBoolQ.

3 Related Work

The degree to which pretrained language models can actually reason is an intensely studied area of research. Helwe et al. [10] give a survey, showing that BERT models tend to rely on shallow heuristics. A number of these approaches rely on creating probing datasets using simple rule-based perturbations of real examples. However, these rules are easy for the model to learn. When models are trained on the probing datasets, the performance often recovers, e.g. [11], [12].

Another closely related area of study is that of adversarial examples. Jia and Liang [13] add unrelated sentences to SQuAD [14] to get models to misidentify the answer. Training on the adversarial examples again recover performance, but only for the variant it is trained on. Recent model-and-human-in-the-loop approaches have resulted in much more difficult datasets by having a human and a model interact until the human produces an adversarial example that tricks the model. Khashabi et al. [15] perturb BoolQ [8] questions, Bartolo et al. [16] use this approach with varying model strengths to create adversarial examples for SQuAD [14], and Talmor et al. [17] use this approach to create a harder CommonsenseQA 2.0 [18]. Unlike other approaches, model performance degrades significantly even when trained on adversarial examples, while human performance remains relatively high.

We are primarily interested in an automated approach with little access to humans or fine-tuned models. Human-in-the-loop approaches may be labor or cost intensive, while model-in-the-loop approaches require many accesses to a finetuned model. The model accessed could share the same pretraining as the one later being evaluated, even if the evaluated model has also been finetuned on the adversarial examples. In contrast with other automated approaches, we attempt to use an algorithm which is not easily reversible. Finally, compared to other approaches, we are interested primarily in decoupling language understanding and reasoning in naturally occurring text from commonsense or world knowledge.

4 Methods

BoolQ [8] examples are comprised of natural questions paired with a relevant passage from Wikipedia, which are labeled either true or false. As the test set is provided without labels, we use the train set and the validation set only with 9.4k and 3.2k examples respectively. We refer to the combined passage and question as a document. In our case, we wish to perturb a document in a way that preserves the label.

In order to preserve the language concepts used in the document, we wish to change the entities contained in the document without changing the structure. We can decompose the document into its words $(x_1, \ldots, x_n)$ and a mapping from the word index $i \in \{1, \ldots, n\}$ to the indices in the document. Thus, we randomize the words x_i, and produce a new document by applying the original mapping to the new words.

However, we cannot simply randomize all the words in the document, since some of which may be structural, e.g. “is” or “the”. We must also preserve some of the relationships between different entities, e.g. a noun and its plural form must be mapped grammatically. Lastly, the randomization procedure should select words that are of the correct category. For example, if we are randomizing a word such as “Canada”, we may be required to choose a different country as a replacement for the resulting document to make sense.

4.1 TF-IDF, FastText, and GMM

We use TF-IDF, fastText [19] word embeddings[1], and Gaussian mixture models (GMM) [9] to group, select, cluster, and randomize entities.

Grouping. We group words if we expect them to change in the same direction, e.g. different forms of the same noun, or strongly related entities. To do this, we give each pair of words a weight g_{ij} which represents whether we need to preserve their relationship during randomization. We note that pairs of words with high prior co-occurrence probabilities, e.g. $P(x_i \in d | x_j \in d)$, should have a higher g_{ij}. We also note that as $\min(n_i, n_j)$ increases, where n_i is the count of word i in the document, g_{ij} *decreases*, since as the words are used more often in a single document, their relationship can be inferred from the document rather than their prior relationship. Thus, we define the weighting

$$g_{ij} = \frac{w_{x_i}^\top w_{x_j}}{\|w_{x_i}\| \|w_{x_j}\|} - \alpha \log(\min(n_i, n_j)) \tag{1}$$

where w_x is the word embedding for the word x, and α is manually tuneable constant. Because morphology is important, we opt to use fastText [19] word embeddings over other word embeddings like GloVe [21]. We use the cosine similarity so that the first term has the appropriate scale, since otherwise the dot product can have issues when some terms are very popular. If a word is out of vocabulary (OOV) or too popular, i.e. above a given `max_doc_freq`, then we set all its weightings to 0. Given a minimum `grouping_cutoff`, we then group all words that are in the same connected component with connections with weight above the cutoff.

Selection. For selecting which words to replace, we note that we wish to replace words that are unique to the given document and which appear many times in the document. Thus, we use TF-IDF (Term

[1]We use the gensim implementation [20].

Frequency-Inverse Document Frequency) for extracting relevant entities. We used the particular formulation

$$\mathrm{tfidf}(t, d, D) = (1 + \log(f_t(d))) \left(1 + \log\left(\frac{1 + |D|}{1 + |\{d' \in D : t \in d'\}|}\right)\right), \tag{2}$$

where t is the term, d is the document, D is the set of documents, and $f_t(d)$ is the proportion of times t shows up in the document d, so that the tfidf weights are scaled appropriately across documents.

We then look at the words with the highest tfidf weights above a given minimum `tfidf_cutoff` and occurrence at least a given `min_term_count`. We select their corresponding groups of words, with a maximum `group_size`, until the total frequency of selected words is greater than `freq_cutoff`.

Randomization. We cluster the selected words across all documents in a test split using their fastText word embeddings. Because of the high dimensionality of the word embeddings, we normalize the vectors and use GMM[2] to cluster the word embeddings into `n_components` clusters, but constrain the covariances to be spherical. We found that PCA would not have reduced the dimension much in this case. We take the best of `n_init` initializations.

Afterwards, we can resample words from these clusters to produce a perturbed document. Note that we use the clusters themselves, and not the Gaussian mixtures to resample. Given the candidate word x_i and sampled word $\tilde{x}_i$, we can perturb all words x_j in a group by performing a word analogy as in [23]. That is, we choose $\tilde{x}_j$ so that $w_{\tilde{x}_j}$ is closest to $w_{x_j} - w_{x_i} + w_{\tilde{x}_i}$ in terms of cosine similarity. We can average these cosine similarities to calculate a similarity score for the perturbed group.

Note that the candidate word and the sampled word may be of different word forms, so we wish to select the candidate word carefully. For a group of words, we find their clusters, pick a random cluster, and sample a word from that cluster. We try each group word in the same cluster as a candidate word to be replaced by the sampled word, and pick the candidate word with the highest similarity score. We check to make sure all the new clusters for the perturbed group match, and if not, we try a different cluster until all clusters have been exhausted, in which case we select the perturbed group with the highest cluster match and similarity score.

As a last step, for any OOV words, we replace them with the word "redacted" in case they relate to any replaced words.

5 Experiments

5.1 Experimental Setup

Datasets. We use the BoolQ dataset [8]. For grouping with fastText, we found empirically that $\alpha = 0.01$, `grouping_cutoff` $= 0.66$, and `grouping_cutoff` $= 0.1$ worked well. For selection, we chose `tfidf_cutoff` $= 5.0$, `group_size` $= 8$, `min_term_count` $= 2$, `freq_cutoff` $= 0.2$. For GMM, we searched `n_components` in the range $[0, 200]$, and found that the Bayesian Information Criterion (BIC) [24] is not minimized in that range. However, we chose `n_components` $= 41$ where the BIC value shows the largest drop in speed of decrease. We choose `n_init` $= 5$.

Training. We trained the base and large sizes of BERT [3] and RoBERTa [4]. We used a softmax classifier with binary output on the first token, which is a special classification token. We train for 10 epochs selecting the model with the best accuracy on the validation set. We use a batch size of 16, the AdamW optimizer [25] with weight decay 0.1, $\beta_1 = 0.9$, and $\beta_2 = 0.98$, and an inverse sqrt learning rate schedule with linear warmup of 400 steps and peak learning rate in [1e-5, 2e-5].

5.2 Perturbation Quality

Grouping. To evaluate the quality of the grouping process, we measure how many documents had a pair of related words placed into different groups. Specifically, we used spaCy [26] to lemmatize every word to its base form, or its *lemma*, and check that words with the same lemma belong to the same group. At `freq_cutoff` $= 0.15$, we found that fastText had 91.8% of documents without lemmatization errors compared with GloVe at 87.9%, justifying our use of fastText over GloVe. At

[2] We use the scikit-learn implementation [22].

`freq_cutoff` $= 0.20$, fastText produced 86.6% of documents without any lemmatization errors. For the main experiments, we filter out any document with lemmatization errors.

Selection. To isolate the selection process, we can consider a variant where we simply mask all selected words using the word "redacted," as we do for OOV words, which we refer to as *MaskedBoolQ*. We trained just RoBERTa-large, which had a validation accuracy of 86.18% on the original BoolQ dataset. When training on MaskedBoolQ generated using fastText vectors with `freq_cutoff` at 0.15 and 0.20, the accuracy dropped to 82.79% and 82.09%, respectively. In a dataset with no statistical regularities, masking all relevant entities should result in a performance exactly equal to the majority class. However, even after over 20% of words were masked, the model surprisingly performed much better than the majority class of 62.74%. This suggests either that we could further increase `freq_cutoff` or that there are statistical cues in the dataset, but we chose to stop at `freq_cutoff` $= 0.20$.

Randomization. After clustering and randomization, the fraction of documents with all perturbed words belonging to the exact same clusters as the original words was 51.9%. However, 93.6% of perturbed words matched the original clusters and the average similarity score was 0.9547. We present a t-SNE [27] visualization of a random sample of 20 out of the 41 clusters in Figure 1.

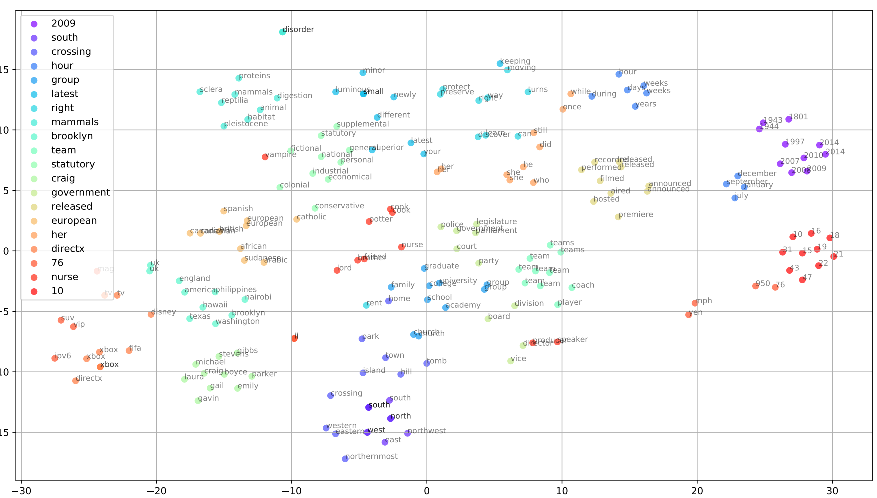

Figure 1: t-SNE visualization of about half of the clusters. The GMM learns to group numbers, years, months, directions, locations, countries, names, nouns, and verbs, among other things.

5.3 Main experiments

Dataset variants. We consider a few variants of the BoolQ dataset, including QBoolQ, where only the question is provided, MaskedBoolQ, where selected words are just redacted, and AltBoolQ, where our full procedure is applied. The performance of the models on these datasets are given in Table 1. The results for BoolQ and QBoolQ roughly match known results [6, 8]. We note in particular that none of the models are able to learn the question-only variant well, considering the majority class is 62.17%, and that all model performances degrade slightly on the MaskedBoolQ and AltBoolQ variants, with the stronger RoBERTa models showing the largest drops in performance.

Model	BoolQ	QBoolQ	MaskedBoolQ	AltBoolQ
BERT-base	74.31	66.06	74.93	72.38
RoBERTa-base	81.01	66.33	77.79	76.21
BERT-large	77.65	66.57	76.89	76.10
RoBERTa-large	86.18	67.31	82.09	82.92

Table 1: Validation set accuracy for the given models trained and evaluated on the given datasets.

AltBoolQ. We run models on BoolQ, AltBoolQ, and BoolQ+ which we use to denote the concatenation of the two. To account for the increase in train set size, we train for only 5 epochs for BoolQ+.

We give the validation set performances on each dataset in Table 2. We see that adding the perturbed examples boosts performance for both BERT models, but not for RoBERTa. Interestingly, many of the models trained on only BoolQ also perform well on AltBoolQ, suggesting that the behaviors they learn are not very reliant on the entities themselves, but on the language and structure.

Model	**Train set**	**BoolQ**	**AltBoolQ**	**BoolQ+**
BERT-base	BoolQ	74.31	71.25	72.87
BERT-base	AltBoolQ	71.01	72.38	71.65
BERT-base	BoolQ+	**76.12**	**74.97**	**75.57**
BERT-large	BoolQ	77.65	75.24	76.51
BERT-large	AltBoolQ	74.04	76.10	75.01
BERT-large	BoolQ+	**77.86**	**77.17**	**77.53**
RoBERTa-base	BoolQ	**81.01**	78.31	**79.74**
RoBERTa-base	AltBoolQ	76.70	76.21	76.47
RoBERTa-base	BoolQ+	79.91	**78.48**	79.24
RoBERTa-large	BoolQ	**86.18**	82.85	**84.61**
RoBERTa-large	AltBoolQ	84.65	82.92	83.84
RoBERTa-large	BoolQ+	85.60	**83.23**	84.48

Table 2: Validation set accuracy for the given models and train sets on the given validation datasets. The best accuracy for each validation set and model are bolded.

5.4 Human Evaluation

To evaluate human performance on the new dataset, the author manually answered 40 examples from both BoolQ and AltBoolQ datasets, scoring 95% and 85% respectively, compared to the published human performance of 89.0% [6]. Because the perturbation can alter the meanings of many words, many AltBoolQ questions become harder, requiring careful reading of the text. Sometimes, the question can be technically unanswerable given just the passage, unless the original topic is known. We give an example in Table 3. We note that the performance of our best model is relatively close to human performance on both datasets.

Passage: Nuclear power in Canada – Nuclear power in Canada is provided by 19 commercial reactors with a net capacity of 13.5 Gigawatts (GWe), producing a total of 95.6 Terawatt-hours (TWh) of electricity, which accounted for 16.6% of the nation's total electric energy generation in 2015. [...]
Question: is there any nuclear power plants in canada
Perturbed Passage: Spring toll in Caribbean – Spring toll in Caribbean is provided by 19 commercial stores with a net capacity of 13.5 Gigawatts (GWe), producing a bad of 95.6 Terawatt-hours (Redacted) of streams, which accounted for 16.6% of the nation's bad stream streams generation in 2015. [...]
Perturbed Question: is there any spring toll plants in caribbean

Table 3: An example passage and question from BoolQ, and the output from our procedure. The procedure fails to identify that "plants" and "reactors" are synonymous, making the perturbed example technically unanswerable.

6 Conclusion/Future Work

By randomizing the entities in the BoolQ dataset, we are able to generate a more difficult dataset AltBoolQ. In some instances, adding examples from AltBoolQ can boost performance on the BoolQ dataset. The closeness in performance of models trained on BoolQ and AltBoolQ also show that language models were able to generalize relatively well, and rely not just on cues from the entities in the text. This could suggest either that the model is actually understanding the text, or that the dataset is too easy. An interesting line of future work would be seeing whether this also applies to other diverse and harder datasets as well. The current perturbation procedure has obvious flaws, and in the future, we could see ways to improve it so that it is more robust. We hope that this leads to more automated procedures for generating hard examples that test pure language understanding.

7 Contributions

Jeffrey Shen worked on all parts of this project.

References

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL `https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf`.

[2] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding with unsupervised learning. 2018.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL `https://www.aclweb.org/anthology/N19-1423`.

[4] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019. URL `http://arxiv.org/abs/1907.11692`.

[5] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL `http://jmlr.org/papers/v21/20-074.html`.

[6] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. *arXiv preprint 1905.00537*, 2019.

[7] Adam Roberts, Colin Raffel, and Noam Shazeer. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5418–5426, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.437. URL `https://aclanthology.org/2020.emnlp-main.437`.

[8] Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of NAACL-HLT 2019*, 2019.

[9] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

[10] Chadi Helwe, Chloé Clavel, and Fabian M. Suchanek. Reasoning with transformer-based models: Deep learning, but shallow reasoning. In *3rd Conference on Automated Knowledge Base Construction*, 2021. URL `https://openreview.net/forum?id=Ozp1WrgtF5_`.

[11] Nora Kassner and Hinrich Schütze. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7811–7818, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.698. URL `https://aclanthology.org/2020.acl-main.698`.

[12] Tom McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1334. URL `https://aclanthology.org/P19-1334`.

[13] Robin Jia and Percy Liang. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1215. URL `https://aclanthology.org/D17-1215`.

[14] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1264. URL `https://aclanthology.org/D16-1264`.

[15] Daniel Khashabi, Tushar Khot, and Ashish Sabharwal. More bang for your buck: Natural perturbation for robust question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 163–170, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.12. URL `https://aclanthology.org/2020.emnlp-main.12`.

[16] Max Bartolo, Alastair Roberts, Johannes Welbl, Sebastian Riedel, and Pontus Stenetorp. Beat the AI: Investigating adversarial human annotation for reading comprehension. *Transactions of the Association for Computational Linguistics*, 8:662–678, 2020. doi: 10.1162/tacl_a_00338. URL `https://aclanthology.org/2020.tacl-1.43`.

[17] Alon Talmor, Ori Yoran, Ronan Le Bras, Chandra Bhagavatula, Yoav Goldberg, Yejin Choi, and Jonathan Berant. CommonsenseQA 2.0: Exposing the limits of AI through gamification. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021. URL `https://openreview.net/forum?id=qF7FlUT5dxa`.

[18] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. 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)*, pages 4149–4158, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1421. URL `https://aclanthology.org/N19-1421`.

[19] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5: 135–146, 2017. ISSN 2307-387X.

[20] Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. `http://is.muni.cz/publication/884893/en`.

[21] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL `https://aclanthology.org/D14-1162`.

[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[23] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013. URL `http://dblp.uni-trier.de/db/journals/corr/corr1301.html#abs-1301-3781`.

[24] Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978. ISSN 00905364. URL http://www.jstor.org/stable/2958889.

[25] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

[26] Matthew Honnibal and Ines Montani. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear, 2017.

[27] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. URL http://jmlr.org/papers/v9/vandermaaten08a.html.