Domain-Specific Japanese ELECTRA Model Using a Small Corpus

Youki Itoh and Hiroyuki Shinnou
Faculty of Engineering, Ibaraki University
4-12-1 Nakanarusawa, Hitachi, Ibaraki 316-8511 Japan
{21nm708a, hiroyuki.shinnou.0828}@vc.ibaraki.ac.jp

Abstract

Recently, domain shift, which affects accuracy due to differences in data between source and target domains, has become a serious issue when using machine learning methods to solve natural language processing tasks. With additional pretraining and fine-tuning using a target domain corpus, pretraining models such as BERT[1] can address this issue. However, the additional pretraining of the BERT model is difficult because it requires significant computing resources.

The efficiently learning an encoder that classifies token replacements accurately (ELECTRA) pretraining model replaces the BERT pretraining method's masked language modeling with a method called replaced token detection, which improves the computational efficiency and allows the additional pretraining of the model to a practical extent.

Herein, we propose a method for addressing the computational efficiency of pretraining models in domain shift by constructing an ELECTRA pretraining model on a Japanese dataset and additional pretraining this model in a downstream task using a corpus from the target domain.

We constructed a pretraining model for ELECTRA in Japanese and conducted experiments on a document classification task using data from Japanese news articles. Results show that even a model smaller than the pretrained model performs equally well.

1 Introduction

A domain shift problem occurs when using machine learning to solve natural language processing tasks, where the training data (source) domain is differs from the test data (target) domain.

Because downstream tasks fine-tune a model, pretrained models such as BERT (Devlin et al., 2019) can deal with the domain shift problem. To improve accuracy, the additional pretraining of BERT using the target domain corpus and fine-tuning of the additional pretrained models have been used recently. However, the additional pretraining of BERT requires considerable computing resources and therefore cannot be performed easily. Furthermore, a large corpus of the target domain to be used is required, but this corpus is often unavailable in reality.

In this paper, we attempted to address the computational efficiency of pretraining in domain shifting using efficiently learning an encoder that classifies token replacements accurately (ELECTRA) (Clark et al., 2020).

ELECTRA is a pretraining model that uses replaced token detection (RTD) to replace the masked language modeling (MLM) used in BERT. In MLM, the model is trained by estimating the [MASK] and replacing some words in the input sentence with [MASK]. However, only 15% of the words in BERT are replaced by [MASK], which is computationally inefficient: thus, RTD is an improvement on BERT.

RTD provides two models: generative and discriminative, which are based on the idea of generative adversarial network (GAN) (Goodfellow et al., 2014). The discriminator is pretrained by deciding if it replaces each token generated by the generator. RTD is computationally more efficient because it can handle all tokens in training, and the ELECTRA model using RTD performs better than the BERT model of the same size.

In this study, we first built a general small-scale ELECTRA model. Thereafter, we constructed a domain-specific ELECTRA model by additional pretraining it using a small corpus of the target domain. Although the constructed domain-specific ELECTRA model is smaller than BERT-base, we confirmed that it outperforms BERT-base in the

[1]Bidirectional Encoder Representations from Transformers

Proceedings of Recent Advances in Natural Language Processing, pages 640–646
Sep 1–3, 2021.
https://doi.org/10.26615/978-954-452-072-4_072

target domain for a document classification task.

2 Method

The corpus used to train pretraining models can be biased(Bai et al., 2020). We can use a pretraining model unique to the domain that the actual system is targeting to increase accuracy in that domain if we construct one. However, most of the current pretraining methods require considerable computational resources to be effective.

In most cases, increasing the amount of computation needed for pretraining, would increase the accuracy of the downstream task; however, when conducting pretraining, considering the accuracy of the downstream task as well as the computational performance is important.

Therefore, herein, we use ELECTRA, that uses RTD, a computationally efficient pretraining method.

ELECTRA models of various sizes evaluated the performance of the downstream tasks considering computational complexity. Experiments on GLUE (Wang et al., 2018), a benchmark for natural language understanding, and SQuAD (Rajpurkar et al., 2016), a benchmark for question answering techniques, were performed. For the same amount of computation, the ELECTRA model outperforms other state-of-the-art natural language processing models. For example, it performs RoBERTa and XLNet with less than 25% of the computational effort.

For further efficiency, ELECTRA-Small, which can be trained in four days on a single GPU, performs better than GPT and requires only 1/30 of the computation.

Building a pretraining model using ELECTRA, which employs RTD, a more computationally efficient pretraining method, and performing additional pretraining on the corpus in the target domain, we can build a model with accuracy comparable to existing pretraining models for document classification tasks with fewer computational resources and less training time; see Figure 1 for an overview.

3 Experiments

3.1 Pretraining Model

We used a program (`run_pretraining.py`) available on the official GitHub [2], free TPU resources on Google Colaboratory (Colab), and Google Cloud Storage (GCS) to build the ELECTRA model with pretraining in Japanese. We can reduce the training time even more than the GPU using the TPU resources on Colab. Furthermore, TPU on Colab can only input and output data via GCS, so using GCS is necessary.

For the pretraining corpus, we used the full text of Wikipedia in Japanese, which is same as BERT from Tohoku University (Tohoku-BERT). We used Mecab-NEologd for the tokenizer as well as for Tohoku-BERT.

To build the training corpus, preprocess the text, create the vocabulary files, and create the TensorFlow dataset for pretraining, the software on Tohoku-official BERT's GitHub[3] was used.

3.2 Model Evaluation

The build model was evaluated based on its success on a document classification task in a small domain. We used the Livedoor-news corpus as the evaluation data for fine-tuning. This is a dataset of Japanese news articles from Livedoor-news published by RONDHUIT Inc.

Each document comprises a URL, creation date, a title, and a body text. Here, we labeled the article body numerically according to its category.

We divide the text of an article into training and test data for each of the nine categories, train a model on the training data, perform a nine-value classification task on the test data to predict the article's category from the text of the article, and assess performance based on the percentage of correct answers.

The numerical labels for each category and number of articles included are shown in Table 1.

3.3 Results

ELECTRA-JP-Small, the model we developed, was pretrained with small size parameters rather than base size parameters, as in Tohoku-BERT. This is because we also consider the computational efficiency of pretraining. In fine-tuning, training is done up to 50 epochs. The trained models are saved for each epoch, and the value with the highest percentage of correct task answers for each model is selected. The results are shown in Table 2.

We perform model comparison experiments using the SentencePiece-based Japanese ELECTRA model (ELECTRA-JP-SentencePiece) released by

[2]https://github.com/google-research/electra

[3]https://github.com/cl-tohoku/bert-japanese

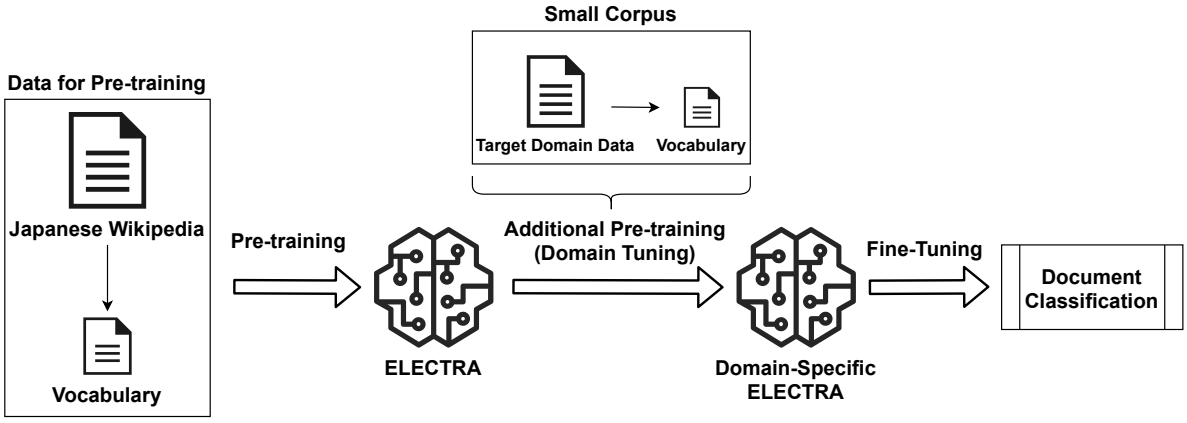

Figure 1: Overview of the method.

Class	Category	Train	Test
0	dokujo-tsushin	87	696
1	it-life-hack	87	696
2	kaden-channel	86	692
3	livedoor-homme	51	409
4	movie-enter	87	696
5	peachy	84	674
6	smax	87	696
7	sports-watch	90	720
8	topic-news	77	616
total		736	5895

Table 1: Numerical labels and number of articles in each category.

Model	Accuracy
Tohoku-BERT	0.8835
ELECTRA-JP-Small	0.8412
ELECTRA-JP-SentencePiece	0.8024

Table 2: Experimental results of fine-tuning. *Accuracy* is the highest percentage of correct answers for each model.

Cinnamon AI Inc. as a guide. This model is a pretrained model with the same parameter size as ELECTRA-Small.

The ELECTRA-JP-Small model provides a higher percentage of correct answers than the ELECTRA-JP-SentencePiece model, (Table 2). However, because the model's parameter size is smaller than the base size, the correct response rate is approximately 4% lower than the Tohoku-BERT model.

3.4 Additional Pretraining

Additional pretraining using this small corpus would enable us to create comparable models; we have been fine-tuning using a domain-specific small corpus and noting the computational efficiency of the ELECTRA models,

Therefore, we experimented to confirm this process. The text of articles from the Livedoor-news corpus, which was used in the previous experiment, was extracted in plain text and used as a single pretraining dataset for the additional pretraining of ELECTRA-JP-Small.

The ELECTRA-JP-Small has already been pretrained for 1 M steps (24 h in training time). We increase the number of steps in this model (ELECTRA-JP-Small-1.25M) by 0.25 M (10 h of training time); see Figure 2 for an overview.

After the additional pretraining, we run the fine-tuning five times on the models, extract the highest correct response rate value from each of the five models, and show the average and highest values in Table 3.

As shown in the table, even with a small parameter size model, we could to confirm that by performing additional pretraining with a small corpus of domain-specific data, we can create an ELECTRA model that outperforms Tohoku-BERT.

3.5 Prediction Time

The ELECTRA model we developed is smaller than the Tohoku-BERT model. This is expected to reduce the training and prediction times during fine-tuning. To confirm this, we measured the training and prediction times for each of the 50 models created during fine-tuning for both ELECTRA-JP-Small and Tohoku-BERT. The results are shown in

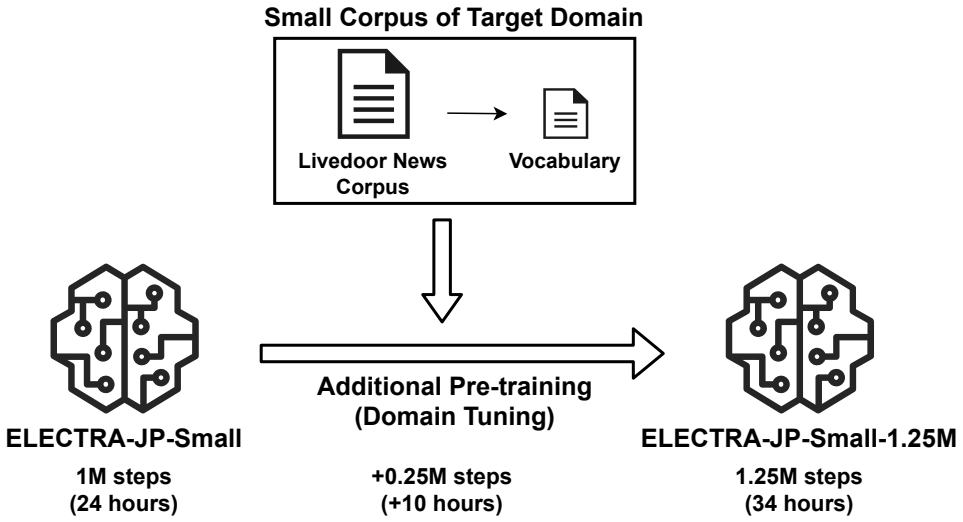

Figure 2: Overview of additional pretraining.

Model	Average	Max
Tohoku-BERT	0.8814	0.8835
ELECTRA-JP-Small-1.25M	0.8834	0.8864

Table 3: Experimental results of fine-tuning after additional pretraining. *Average* is the average of the values obtained by running fine-tuning five times for each of the five models with the highest percentage of correct answers. *Max* is the highest value of the five models' correct responses.

Figure 3.

The average training time for each BERT model was approximately 61.7 s, and for each ELECTRA model was approximately 14.4 s. Moreover, each model in BERT took an average of 160.4 s to predict, and in ELECTRA took on average of 33.6 s. Summing up the time taken by each of the 50 models, the training time for BERT was 51 min 25.1 s and the prediction time was 2 h 13 min 39.7 s, whereas the training time for ELECTRA was 11 min 58.7 s and the prediction time was 27 min 59.7 s. From these results, it can be seen that ELECTRA-JP-Small can train the model in approximately 1/4 the length and predict in approximately 1/5 the length than Tohoku-BERT.

3.6 Model Size

Table 4 shows the predictions of the pretraining time up to 1 M steps using TPU for different model sizes.

The results in Table 2 are not an exact comparison of model performance because of the performance difference due to the parameter size of the pretraining model. Even with one TPU resource, building an ELECTRA model with the same parameter size as Tohoku-BERT requires approximately a week of training time. As pretraining requires huge computational resources, the larger the model size, the more difficult it becomes to build a pretrained model. The ELECTRA model, which is more parameter efficient for smaller models, the performance was reasonably good even for small size. However, this does not go far enough to override the performance of the different model sizes. To confirm the difference in model performance more accurately, constructing pretraining models of the same size is necessary.

Model	Time
ELECTRA-JP-Small	1d 22hs
ELECTRA-JP-Base	7d 1h

Table 4: Pretraining time for models up to 1M steps. The time in the table is the predicted pretraining time for the model up to 1M Steps.

3.7 Effects of Additional Pretraining

From the results in Tables 2 and 3, it is clear that the pretraining model with a small corpus of domain-specific data before the downstream task can im-

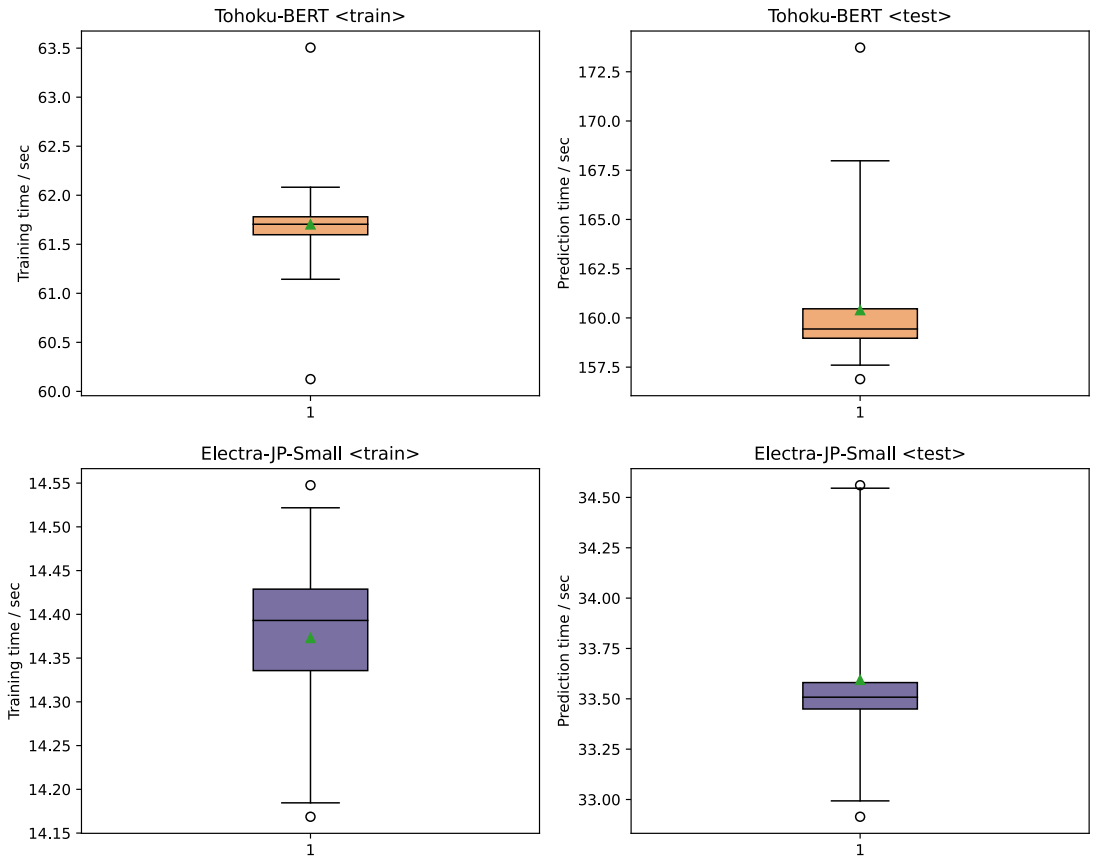

Figure 3: Box plots of training and prediction times for Tohoku-BERT and ELECTRA-JP-Small.

prove the model's accuracy after fine-tuning.

The larger the parameter size of the model, even for BERT and ELECTRA, the more time it takes to pre-train the model, and thus the more difficult it is to perform additional pretraining. However, for small models, the time required for additional pretraining is much shorter than for base-size models. Therefore, additional pretraining of the ELECTRA-Small model with a small corpus of domain-specific models can achieve better performance with less computation.

4 Related Work

4.1 Masked Language Modeling

MLM masks a certain percentage of words in the input sentence (usually approximately 15%). The masked words are replaced with other words or special tokens, such as [MASK], and the model is assigned to predict the masked words. BERT with such a pretraining method outperforms the conventional language model on many of the downstream tasks. However, because the MLM methods is used to predict masked words, only 15% of the masked words in each input sentence are used to train the model. Therefore, pretraining a model using MLM requires considerable computational resources. additionally, the token [MASK], which represents the mask, exists only during pretraining and does not appear during fine-tuning. This mismatch of [MASK] tokens between pretraining and fine-tuning slightly degrade the performance of MLM pretraining models.

4.2 Replaced Token Detection

ELECTRA employs a new pretraining method called RTD to improve the weaknesses of MLM. It learns a bidirectional model like MLM while learning from all input sentences like a conventional language model.

Based on the idea of GAN, RTD trains the generative model to distinguish between "real" and "fake" input words. Rather than collapsing the input sentence by replacing a word with [MASK], as BERT dose, RTD collapses the input sentence by replacing it with a "fake" word that is false but plausible compared to the original sentence. Thereafter, the discriminative model is trained to take the

collapsed sentence as input and to predict which words have been replaced compared to the original input sentence.

The generator model can be any model that produces token output distributions, but a previous study (Clark et al., 2020) used a small MLM model trained concurrently with the discriminator model, i.e., the BERT model with a small hidden layer scale. Although the relationship between the generator and discriminator is similar to the structure of GAN described above, applying GAN to the text domain is not easy. Therefore, the generator is trained using the maximum likelihood to predict masked words rather than adversarial. If the generator correctly predicts the original word of the masked word, the word is labeled as the original word.

The generator and discriminator share the same input word embeddings, and after pretraining, the generator is removed and only the discriminator is adjusted in the downstream task. Both the two models use the neural transformer architecture.

4.3 Analyzing the Efficiency of MLM

MLM is considered inefficient, and to confirm this, Clark et al. (2020) set up and experimented with three models. The performance of ELECTRA is improved by defining the loss for all input tokens instead of only partially. BERT's performance is slightly impaired by the discrepancy between pretraining and fine-tuning in the [MASK] token. BERT has already been replaced with random tokens as a measure to improve this mismatch; however, even this measure is not sufficient. Consequently, ELECTRA is computationally more efficient than BERT owing to more efficient tokens and less mismatch during fine-tuning (use of MASK symbols). The improvement in ELECTRA can also be attributed to other factors than fast learning. ELECTRA outperforms BERT by a wider margin when the model size is smaller, and it converges perfectly. Although more analysis is needed, ELECTRA is generally considered more parameter efficient than BERT.

Because ELECTRA-JP-Small and Tohoku-BERT model sizes are different, we did not confirm the exact difference in computational efficiency, but if the two Japanese models of the same size are trained together, ELECTRA will likely produce better accuracy.

4.4 Unsupervised Domain Adaptation

Models that generate contextualized word embeddings, such as ELMo and BERT, perform well across natural language processing tasks when pretrained on large unlabeled corpora. Although these models use corpora such as Wikipedia or news texts for training, it is unclear whether this approach is effective when the domain and writing style of the target text differ significantly from the pretrained corpus. However, fine-tuning the distributed representation using the text in the target domain improved the performance (Han and Eisenstein, 2019).

In the related research, two texts were tested as target domains: Early Modern English and Twitter. Both are different from the existing pretrained corpora, but the proposed method provides substantial improvements over the BERT model.

We use the computationally efficient ELECTRA model for fine-tuning the target domain instead of BERT to improve the accuracy in downstream tasks and increase the computational speed using a smaller model.

4.5 Domain/Task Tuning

Contextualized word embeddings may be ineffective for tagging tasks when the target domain is different from the pretrained corpus. This is especially serious for unsupervised domain adaptation because the labeled data may differ significantly from the target text. To address this problem, a method of the AdaptaBERT model for unsupervised domain adaptation was proposed.

Specifically, the following two approaches have been applied.

Domain Tuning Unsupervised tuning of the BERT language model using text from the target domain. For example, BERT trained on Wikipedia can be retrained using Twitter text.

Task Tuning A way to tune BERT using teacher data. For example, for named entity recognition, this would be to train the entire model including BERT using CoNLL 2003.

The four experimental settings are defined by the combination of domain and task tunings.

- Frozen BERT
- Task-tuned BERT
- AdaptaBERT

- Fine-tuned BERT

Frozen BERT is a method that uses BERT as a feature extractor without training it. Task-tuned BERT is a method that BERT is trained using supervised data from the source domain. AdaptaBERT is a method in which BERT is tuned using unsupervised data from the target domain and then the model is trained using supervised data from the source domain. The last one, Fine-tuned BERT, is a method to train the entire model including BERT using data from the target domain.

The experiment evaluates part-of-speech tagging in the Penn Parsed Corpus of Early Modern English (PPCEME) and named entity recognition in the Workshop on Noisy User Text (WNUT) 2016.

For part-of-speech tagging, used the Penn Treebank (PTB) corpus of 20th century English as the source domain corpus and PPCEME as the target domain corpus. PTB corpus is for modern English, and the PPCEME corpus is for 15th to 17th century English.

For named entity recognition, used Conference on Natural Language Learning (CoNLL) 2003 as the source domain corpus and WNUT 2016 as the target domain corpus. CoNLL 2003 is for news, and WNUT is for Twitter.

The results of part-of-speech tagging and named entity recognition show that using Domain Tuning to train BERT on a corpus of the target domain improves performance. Even if no large amount of labeled data in the target domain is unavailable, improving performance by simply tuning the pretraining model using unsupervised data from the target domain is practical.

In our experiments, we assume a situation where a large amount of labeled data in the target domain cannot be secured. We are exploring how much Domain Tuning can improve the performance of ELECTRA-Small under this situation.

5 Conclusion

In this paper, focusing on the computational efficiency of the ELECTRA model and its good performance at scale, we constructed a domain-specific pretrained ELECTRA model by additional pretraining it using a small corpus of the target domain.

Although the constructed ELECTRA model is smaller than the Tohoku-BERT model to be compared, it achieved higher performance than Tohoku-BERT in document classification in the target domain.

In future work, we would like to construct a pretraining model of the same size to compare the performance of the two models more rigorously.

Acknowledgment

The research was supported by JSPS KAKENHI Grant Number JP19K12093 and ROIS NII Open Collaborative Research 21FC05.

References

Jing Bai, Rui Cao, Wen Ma, and Hiroyuki Shinnou. 2020. Construction of domain-specific distilbert model by using fine-tuning. In *2020 International Conference on Technologies and Applications of Artificial Intelligence (TAAI)*, pages 237–241. IEEE.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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. Association for Computational Linguistics.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27, pages 2672–2680. Curran Associates, Inc.

Xiaochuang Han and Jacob Eisenstein. 2019. Unsupervised domain adaptation of contextualized embeddings for sequence labeling. 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)*, pages 4238–4248, Hong Kong, China. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. *CoRR*, abs/1606.05250.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.