GAN-LM: Generative Adversarial Network using Language Models for Downstream Applications

Dae Yon Hwang Yaroslav Nechaev Cyprien De Lichy Renxian Zhang
Amazon Alexa AI
{dyhwang, nechaey, cllichy, renxiz}@amazon.com

Abstract

In this work, we investigate Data Augmentation methods to improve the performance of state-of-the-art models for four different downstream tasks. Specifically, we propose Generative Adversarial Network using Language Models (GAN-LM) approach that combines a deep generative model with a pre-trained language model to produce diverse augmentations. We compare the GAN-LM to various conventional methods in non-contextual- and contextual-levels on four public datasets: *ZESHEL* for zero-shot entity linking, *TREC* for question classification, *STS-B* for sentence pairs semantic textual similarity (STS), and *mSTS* for multilingual sentence pairs STS. Additionally, we subsample these datasets to study the impact of such augmentations in low-resource settings where limited amounts of training data is available. Compared to the state-of-the-art methods in downstream tasks, we mostly achieve the best performance using GAN-LM approach. Finally, we investigate the way of combining the GAN-LM with other augmentation methods to complement our proposed approach. The developed code for reproducibility is included in the supplementary material.[1]

1 Introduction

Nowadays, the availability of large unsupervised corpora and computational resources has led to development of large language models (LMs) that are now employed across a wide variety of natural language processing (NLP) tasks including but not limited to entity linking (EL), text summarization, question classification (QC) and semantic textual similarity (STS). While such models can sometimes work well for tasks where little (few-shot) or no (zero-shot) supervised data is available, the performance loss in such low-resource settings can be substantial compared to their high-resource counterparts. This gap is even larger for low-resource languages. Thus, scientists in both industry and academia still have to rely on a multitude of methods Hedderich et al. (2021), such as Data Augmentation (DA), to yield sufficient levels of performance on their low-resource tasks.

DA allows to artificially increase the size of a dataset by generating additional synthetic examples from the existing ones. A large amount of diverse training data is important to ensure the generalization of a model but it is not always possible to collect due to cost and time constraints or lack of target language data and task experts. To mitigate this issue, DA can be used to improve performance.

In this work, we test a wide variety of DA approaches, both the ones found in the literature as well as our own approach, on four different tasks: Zero-shot EL with *ZESHEL* dataset, QC with *TREC* database, STS with *STS-B* dataset, and multilingual STS with *mSTS* database. Two different levels of augmentations are considered: (1) Non-contextual, or word-level, and (2) Contextual, where full sentence is considered for DA. To further highlight the impact of different DA approaches, we produced low-resource versions of the above-mentioned tasks by subsampling training sets and removing rich textual contexts where applicable. We propose a novel Generative Adversarial Network using Language Models (GAN-LM) which employs GAN with Wasserstein distance to improve the stability of training and uses the pre-trained LM for generating synthetic textual data to extend its usability. We also introduce tunable thresholds and a decoding method to control the diversity and lexical similarity of synthetic data to mitigate the mode collapse problem in GAN. Compared to other DAs, GAN-LM employs an adversarial training with the offered data in each task to learn the characteristic of it which generates suitable synthetic data to aid in downstream tasks (covered in Section 5.6). Even if we used pre-trained LM in GAN-LM, we do not use its

[1]https://github.com/amazon-science/data-augmentation-for-entity-resolution

generation ability (e.g. paraphrase, text generation) for downstream tasks. To complement our approach, we mix GAN-LM with other DAs (e.g. Back-translation, GPT) to enhance further in low-resource languages and limited entity linking task. The source code used to train the GAN model and produce augmentations listed in this paper is publicly released and attached with a paper.

2 Related Work

Originally, DAs for NLP relied on synonyms to increase diversity and dataset size. Synonyms could be found in various resources like WordNet Miller et al. (1990) and PPDB Ganitkevitch and Callison-Burch (2014). In Wang and Yang (2015), they considered word embedding with K-Nearest-Neighbor (KNN) and cosine similarity to search and substitute similar words. Other pre-trained word embeddings such as Word2Vec Mikolov et al. (2013), GloVe Pennington et al. (2014) and fastText Bojanowski et al. (2017) have been leveraged for that purpose. Furthermore, the authors in Wei and Zou (2019) generated synthetic texts by changing the words through synonym replacement or random insertions, substitutions and deletions where Shou et al. (2022) include the abstract meaning representation graph along with it for STS task. In Pruthi et al. (2019), the authors simulated spelling mistakes by random insertions, substitutions, character swaps and deletions to enhance the robustness of the model for sentiment analysis. Also, punctuation as DA was considered in Karimi et al. (2021) for QC task. Later, back-translation with Neural Machine Translation (NMT) was employed to generate variations of target words Sennrich et al. (2016).

More emerging techniques for DA are using deep neural networks which mostly use autoregressive language model to predict words from a given context, e.g. GPT-2 Radford et al. (2019), XLNet Yang et al. (2019) and BART Lewis et al. (2020) which have been used for DA in diverse applications such as question-answering, text classification and machine translation. Using LMs, KNN-based DA with knowledge distillation Kamalloo et al. (2021) is considered for QC task. There are also works related to the adversarial learning to understand their effects on language models. Alzantot et al. (2018) proposed a black-box population-based optimization to generate the imperceptible adversarial examples to fool the models. In Zhang et al. (2019a), they considered Metropolis-Hastings attack to generate the adversarial examples which were tested in terms of attack and training.

However, there are relatively few works using GANs for text generation even if it is one of the most notable approaches in other domains Antoniou et al. (2017). In Kusner and Hernández-Lobato (2016), the authors used a GAN model with Gumbel-Softmax to have a differentiable sampling distribution approximating a categorical one. In Subramanian et al. (2017), diverse GANs with recurrent and convolutional architectures were evaluated for text augmentation at word and character-levels. Yu et al. (2017) proposed a sequence GAN with reinforcement learning to address the problem of assessing a partially generated sequence. Another work in Nie et al. (2018) developed a GAN model consisting of relational memory-based generator, the Gumbel-Softmax relaxation, and multiple embedded representations in the discriminator. In Golovneva and Peris (2020), authors explored a data generation for the bootstrapping of a new language and the handling of low-resource features using a sequential GAN. Croce et al. (2020) used the fine-tuning of BERT with unlabeled data in a generative adversarial setting to reduce the time consuming of annotating the data but did not extend to the DA application. Similarly, Thakur et al. (2021) use the cross-encoder to label the new inputs for training a bi-encoder model. Marek et al. (2021) focus on out-of-domain data generation with a sequential GAN to build the robust dialog system. Compared to these works, GAN-LM combines a large pre-trained model and GAN with tunable thresholds to suitably control the diversity and similarity of generated data and it was tested on various downstream tasks to assess its generalizability. Also, we can use any pre-trained LM on top of the GAN part which extends its applicability to various tasks. To highlight the effectiveness of DA, the low-resource settings are investigated in Shi et al. (2021) and Hedderich et al. (2021) where we mainly investigate the different size of training set and suggest a way to define the optimal size of augmented data.

3 Data Augmentations

3.1 Non-Contextual-Level Augmentation

In this work, we utilize four augmentation approaches as non-contextual-level. ***Lexical:*** We use WordNet Miller et al. (1990) to replace each word in the original text with a synonym. ***Spelling:*** We generate alternate texts from common misspellings of the original words Coulombe (2018). ***Character:*** Here, we randomly change characters in the original tokens with four different ways: Insertions, substitutions, swaps and deletions Pruthi et al. (2019). For lexical, spelling and character-based methods, we use the implementation in *nlpaug*[2] with 10% replacement. ***Token-LM:*** To understand the effectiveness of GAN part in GAN-LM, we consider pre-trained LMs solely. To generate the synthetic data: (1) Use LM to get token embeddings for input text and (2) perform nearest neighbor search for each token to find alternate tokens that meets the similarity thresholds. We search the synthetic tokens which satisfied these thresholds to balance the analogy and diversity, compared to the original token. The similarity thresholds are defined empirically (e.g. Table 5). We did not insert the noise on the input embedding as GAN-LM since the generated data is far from the original one.

3.2 Contextual-Level Augmentation

To extend our work, we experiment three methods as contextual-level augmentation. ***Text Generation:*** This is a typical auto-regressive generation which uses the original text as the initial context and extends it Yang et al. (2020). For this, we employ GPT-2 Radford et al. (2019) and OPT Zhang et al. (2022) for English-based datasets, and mGPT Tan et al. (2021) for multilingual dataset. ***Paraphrase:*** This augmentation transforms a sentence with similar semantic meaning but a different syntactic form where we consider the fine-tuned T5 model Raffel et al. (2020) on Google PAWS Zhang et al. (2019b) for English-based tasks and Prism model Thompson and Post (2020a,b) for multilingual-based task. ***Back-translation:*** It is a process of retranslating content from the target language back to its source language to generate a sentence variant. For this augmentation, we employ multiple pre-trained neural translation models trained on OPUS data Helsinki-NLP (2023) with *nlpaug*.

3.3 Generative Adversarial Network

GAN is basically coming from the adversarial learning which aims to trick the model by providing deceptive input. GAN targets to correctly classify both unmodified and adversarial examples to receive the rewards. It consists of two neural networks, generator and discriminator, where each of them tries to outplay the other. The goal of generator is to artificially manufacture outputs that could be hard to distinguish from real data. The discriminator is similar to the usual classification model that aims to differentiate between real and synthetic data from generator. Using GAN, we target to achieve eminent performances with only offered train set in each downstream task.

Specifically, we considered a WGAN-GP Gulrajani et al. (2017) which uses the Wasserstein distance as loss to capitalize on the probability distributions from fake and real data rather than labeled samples. Compared to the vanilla GAN, it is robust to vanishing gradient and mode collapse through smoother gradient updates from its loss functions.

3.4 GAN-LM

To extend the usability of GAN in NLP domain, we propose GAN-LM which combines GAN with pre-trained LM regardless of non-contextualized and contextualized models. In this work, we focus on the latter one which promises the better result. Loss function of GAN-LM is covered in Equation (1) and its structure is shown in Appendix.

$$\begin{aligned}
&R = LM_{encoder}(\text{Input Text})\\
&\epsilon \sim \text{Uniform}(0,1),\ \eta \sim N(0,1)\\
&F = G(R+\eta), \quad \hat{F} = \epsilon \cdot R + (1-\epsilon) \cdot F\\
&L_d = D(F) - D(R) + \lambda \cdot (||\nabla_{\hat{F}} D(\hat{F})||_2 - 1)^2\\
&L_g = -D(F)
\end{aligned} \tag{1}$$

where $LM_{encoder}$ is the encoder of LM to generate the embedding of input text for augmentation. ϵ and η are random numbers from the uniform and Gaussian distributions respectively. Also, R is the real embedding and F is the fake embedding generated from the generator, $G(\cdot)$. $\hat{F}$ is weighted embedding from real and fake embeddings. $D(\cdot)$ means the discriminator output for embeddings. L_d and L_g refer to the loss functions of the discriminator and generator respectively. $D(F)-D(R)$ describes the 1-Wasserstein distance and $\lambda \cdot (||\nabla_{\hat{F}} D(\hat{F})||_2 - 1)^2$ is called gradient penalty used for mitigating the vanishing gradient where we use $\lambda = 10$ based on the suggestion in Gulrajani et al. (2017).

Figure 1 illustrates the flowchart of the overall algorithm in GAN-LM. First, we generated embeddings for each input text to serve as a training set for the GAN. For embeddings, we use the pre-

[2]https://github.com/makcedward/nlpaug. MIT License.

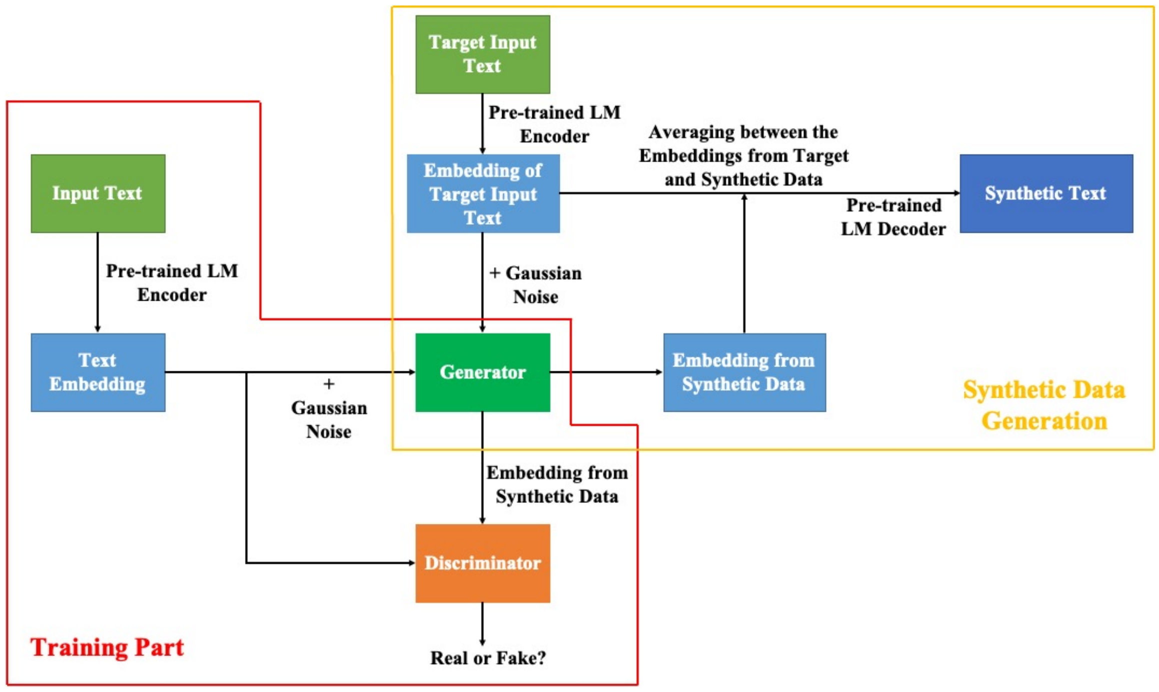

Figure 1: GAN-LM with pre-trained LM. Before decoding from LM, we additionally consider the low and high thresholds for similarity matching between the averaged synthetic embedding from GAN-LM and the candidate embedding from defined dictionary in pre-trained LM to control the diversity and lexical similarity of synthetic text.

trained BART-base Lewis et al. (2020) and mBART-large-50 Tang et al. (2020) as encoder and decoder according to the empirical results (e.g. Table 5). However, GAN-LM can be applied with other approaches that can encode arbitrary text into embeddings and comprise of a well-defined dictionary to map generated embedding back to tokens. Using these transformers, we can decode the synthetic embedding into a text which can be different from the original input. We fix the sentence lengths to 27, 25, 36, 24 tokens for *ZESHEL, TREC, STS-B, mSTS* datasets respectively, which cover 99% of the data for each dataset. To express the text into an input data for GAN training, we stacked each token as the dimension with padding the remainder of the input with zero values (more covered in Appendix).

Figure 1 shows the steps of the algorithm in GAN-LM. In training part (red area), we encode the input text into embeddings using $LM_{encoder}$, then we add Gaussian noise on top and input resulting embeddings to the generator. Next, the generator produces synthetic embeddings which should resemble real ones and feeds those to the discriminator which tries to distinguish between real and synthetic embeddings. In synthetic data generation pipeline (yellow area), we feed the target text, for which we want to generate and alternate form, to the encoder and add Gaussian noise to that embedding. The generator will produce the synthetic embedding for that target text and then we average the original and synthetic embeddings to maintain the structure of original text. To decode, we perform nearest neighbor search for each token using those generated synthetic embeddings. Finally, we introduce upper and lower thresholds on similarity to select tokens that are diverse yet still possess similar semantics compared to the original and don't accidentally change the meaning of the input text. The augmented tokens are selected randomly from tokens that meet those thresholds with the bias towards tokens for high similarity score. Since BART's vocabulary consists of subword units, GAN-LM is able to come up with new valid-looking words that were never in the original training set (see Table 4). To clarify, we employ the pre-trained LM for tokenization and detokenization in GAN-LM but we do not use its text generation methods (e.g. paraphrase) for downstream tasks.

4 Datasets and Employed Models

We experiment with four different downstream tasks where (1-3) are English-based databases and (4) is multilingual-based dataset: (1) *ZESHEL* – a zero-shot learning dataset for EL, (2) *TREC* – a text retrieval dataset for QC, (3) *STS-B* - an integrated version of STS tasks for measuring the semantic similarity between two sentences, and (4) *mSTS* - multilingual version of STS task.

ZESHEL introduced by Logeswaran et al. (2019) is based on Wikia where there are non-overlapping domains in train/validation/test sets to simulate zero-shot learning. For this task, we employ BLINK's Wu et al. (2019) bi-encoder model from scratch. *TREC* shown in Li and Roth (2002) is collected from Hovy et al. (2001) where questions were manually created with 50 fine class labels. For this application, we use fine-tuned BERT-Tiny Turc et al. (2019) with training data in *TREC*. *STS-B* covered in Cer et al. (2017) includes news headlines, image captions and user forum posts. In each sentence pair, semantic similarity labels are provided by a number between 0 and 5. For this task, we use SentenceTransformers Reimers and Gurevych (2019) from scratch using the mean pooling layer with XLM-RoBERTa Conneau et al. (2020). *mSTS* introduced in Cer et al. (2017); Reimers and Gurevych (2020) has sentence pairs in different languages with semantic similarity scores between 0 and 5. For train set, we used the offered monolingual pairs of AR-AR, ES-ES and the translated sentences of ES-ES into EN, DE, TR, FR, IT, NL using Google Translator since we do not have monolingual pairs for them. The provided EN-EN dataset was eliminated from train set since most cross-lingual datasets were made from translating one sentence of EN-EN Reimers and Gurevych (2020). All the cross-lingual pairs are considered as test set. For this application, we employ the mean pooling of outputs for the pre-trained multilingual BERT (mBERT) Devlin et al. (2019) with fine-tuning from train set.

5 Results and Discussion

5.1 Experimental Setting

For all downstream tasks, in addition to the original size, we construct a low-resource version (i.e. limited train set) to highlight augmentation impact. In addition, *ZESHEL* contains rich textual context for both entity mentions and catalog entities, which provide additional information for EL. To isolate the impact of DA approaches, we test model performance with and without those contexts.

For augmentation, in *ZESHEL* we consider both the entity and mention to generate synthetic data, in *TREC* we synthesize an alternate question sentence, in *STS-B and mSTS* we generate an alternate sentence from one of the pair. To build the GAN-LM, we use pre-trained BART and mBART with 0.3-0.7 and 0.5-0.9 similarity thresholds respectively which give a good diversity of generated data while remaining close to the original semantics. The thresholds are decided from empirical results such as Table 5. Also, the size of augmentation is determined from the empirical results in validation set (e.g. Table 6) where we cover the optimal size in this work. Compared to other tasks, we additionally fine-tune GAN-LM with a target language (e.g. AR-AR) in *mSTS* after training with multilingual sentences to boost the quality of synthetic data.

In all tasks, we use the same target metrics as found in the literature. For *ZESHEL* we use recall@k, for *TREC* F1 score, for *STS-B and mSTS* the spearman's rank correlation (SRC) between the cosine similarity of sentence pairs embeddings and ground-truth labels. In all experiments, we retrain target model 3 times with different seeds and report average results with 95% confidence interval (CI). Finally, the computational cost for GAN-LM is covered in Appendix where it takes a longer training time compared to non-contextual-level methods, and comparable time to contextual-level approaches. However, GAN-LM promises the better performances in most cases and utilizes LMs without fine-tuning for generation purpose.

5.2 Results in Entity Linking

Table 1 shows the results for *ZESHEL*. In this application, we target to find the generalized augmentations for zero-shot learning task. In low-resource cases, the amount of train set with augmentation is 5K generated from 1K baseline while in full training data, the size of the training set after augmentation is 69K from 49K baseline.

Overall, improvements after augmentation in normal case are lower than for the low-resource scenario which confirms the importance of augmentation in the limited data setting. Including synthetic data can have an effect of inferring the unseen data which might have a different distribution from train set. Few training set samples in low-resource mean insufficient variation of data to help the models, especially high capacity ones, generalize well. Thus, data augmentation often improves more in low-resource scenarios, compared to normal case.

When we consider scenarios without context, we can see that there are large improvements in performance using augmentations, especially

Table 1: Recall values in *ZESHEL* with 95% CI. Baseline describes the performance of model without augmentation and change denotes the performances against baseline in absolute term. In each scenario, bold means the best results and underline denotes the proposed methods.

Scenarios	Type	R@1	R@8	R@32	R@64	CI	Change
Normal without context	**GAN-LM -GPT**	**28.91%**	**54.83%**	**64.77%**	**69.38%**	1.71%	**7.94%**
	GAN-LM	24.2%	48.96%	60.85%	66.16%	**1.65%**	3.51%
	GPT	28.32%	54.14%	63.31%	67.46%	1.89%	6.77%
	OPT	27.54%	53.28%	62.81%	67.15%	1.89%	6.16%
	Paraphrase	22.1%	46.89%	59.1%	64.73%	2.03%	1.67%
	Back-Translation	20.7%	44.77%	57.13%	62.99%	2.06%	-0.14%
	Token-LM	21.33%	45.52%	57.55%	63.29%	1.83%	0.39%
	Char	22.11%	46.36%	58.5%	64.07%	4.38%	1.22%
	Spel	21.52%	45.76%	58.22%	63.88%	2.25%	0.81%
	Lexical	20.67%	44.8%	57.23%	62.91%	2.01%	-0.13%
Low-resource without context	**GAN-LM -GPT**	**25.25%**	**50.94%**	**59.9%**	**63.8%**	2.3%	**15.11%**
	GAN-LM	18.67%	42.43%	55.21%	61.03%	**1.97%**	9.47%
	GPT	22.52%	47.52%	58.23%	62.62%	2.37%	12.86%
	OPT	19.76%	45.07%	57.06%	61.82%	2.33%	11.07%
	Paraphrase	17.83%	41.16%	53.79%	60%	2.41%	8.33%
	Back-Translation	16.14%	37.71%	50.63%	56.82%	2.84%	5.46%
	Token-LM	15.86%	36.9%	49.98%	56.2%	2.9%	4.87%
	Char	16.52%	37.91%	51.34%	57.53%	2.67%	5.96%
	Spel	16.11%	37.44%	50.63%	56.87%	3.88%	5.4%
	Lexical	15.56%	36.67%	49.9%	56.01%	2.24%	4.67%
	Baseline - Low	12.4%	31.24%	44.65%	51.16%	3.09%	-
	Baseline - Normal	20.57%	44.89%	57.56%	63.13%	1.92%	-

Scenarios	Type	R@1	R@8	R@32	R@64	CI	Change
Normal with context	**GAN-LM**	**39.13%**	**66.45%**	**76.3%**	**79.98%**	**0.65%**	**1.23%**
	GPT	37.36%	65.31%	74.78%	78.65%	1.54%	-0.21%
	OPT	37.63%	65.37%	74.88%	78.77%	0.93%	-0.08%
	Paraphrase	37.88%	65.35%	74.94%	78.7%	0.76%	-0.02%
	Back-Translation	37.73%	65.26%	74.95%	78.73%	1.25%	-0.07%
	Token-LM	37.53%	64.58%	74.49%	78.41%	1.27%	-0.49%
	Char	37.53%	64.68%	74.6%	78.56%	1.37%	-0.4%
	Spel	37.27%	64.42%	74.42%	78.38%	1.19%	-0.62%
	Lexical	37.49%	64.86%	74.89%	78.66%	1.66%	-0.27%
Low-resource with context	**GAN-LM**	**23.93%**	**49.79%**	**61.5%**	**66.75%**	**1.29%**	**3.71%**
	GPT	21.57%	47.75%	59.75%	64.69%	2.05%	1.66%
	OPT	22.84%	47.99%	60.47%	65.38%	1.68%	2.39%
	Paraphrase	20.13%	45.59%	58.36%	63.62%	1.75%	0.14%
	Back-Translation	17.6%	42.25%	54.86%	60.84%	1.98%	-2.9%
	Token-LM	13.76%	35.95%	48.64%	54.97%	1.62%	-8.45%
	Char	14.92%	38.11%	51.17%	57.35%	2.85%	-6.4%
	Spel	19.46%	44.46%	56.85%	62.54%	4.71%	-0.96%
	Lexical	17.59%	41.68%	54.03%	60.18%	2.62%	-3.41%
	Baseline - Low	20.92%	45.19%	57.63%	63.39%	1.59%	-
	Baseline - Normal	37.93%	65%	75.08%	78.95%	1.19%	-

with contextual-level, and GAN-LM mostly outperforms others, except for GPT and OPT. In this case, EL model has been trained on only entity in train set to infer the entity with its contexts in test set. Thus, it can be beneficial to use the augmented data with additional descriptions to imitate the context of it which can be done by GPT and OPT. We further investigate the augmentation from a combination between GAN-LM and GPT, called GAN-LM-GPT. In this approach, GAN-LM generates alternate forms from the original inputs at the token-level and GPT adds new textual content after that. We observe improvements after combinations of both methods, especially in the low-resource case. Therefore, we can also consider GAN-LM-GPT augmentation when train data is limited without additional contexts in entity linking (EL) task since it helps to include the diverse variations in test set which cannot be covered by the considered train set. For scenarios with context, most augmentations, especially with non-contextual-level, decrease the performance since synthetic data from these approaches could be less related to the available contexts which could be harmful to EL. However, GAN-LM has tunable thresholds to control the diversity and similarity of synthetic data which finally promises the improvements. In conclusion, we observe that GAN-LM and its complement, GAN-LM-GPT, are the best choices for EL task whether in low-resource or normal setting. In *ZESHEL*, domains in test set are not overlapped with the ones in train set, which confirms that GAN-LM is fairly compared with other augmentations.

5.3 Results in Question Classification

Now, we test the influence of DAs for question classification (QC) task covered in Table 2 left side. In this task, we need label-invariant augmentations to improve the performance. The size of training data for augmentations is 1K from 109 baseline in low-resource and 8K from 2K baseline in half-train set case. Interestingly, the improvements after augmentations in both scenarios have a similar pattern: Contextual-level augmentations outperforms the non-contextual ones, except for spelling and lexical (only for low-resource) while GAN-LM is always the best performing approach. In addition, the improvements in half-train set scenario are higher than the ones in the low-resource. From our investigation, the result without augmentation in half-train set is considerably worse than the one in normal case (i.e. 8.84% F1 difference), meaning the effect of augmentation can be huge in half-train set to improve further. Also, adding synthetic data on model can be noticed as inferring the possible variations in test set which needs some degree of real traffic from original data to suitably utilize the augmented data. Still, GAN-LM works the best in

Table 2: F1 and SRC values in *TREC* and *STS-B* with 95% CI. Here, we did not cover the normal case for augmentation since we already achieve the better or similar performance with half-train set, compared to full training set without augmentation (i.e. Baseline - Normal). In normal scenario, GAN-LM gives 34.28% F1 score in *TREC* and 79.84% SRC in *STS-B*. Denotations are identical as Table 1.

Question Classification in *TREC*

Scenarios	Type	F1	CI	Change
Half-train set	**GAN-LM**	**32.14%**	2.23%	**16.01%**
	GPT	29.16%	2.66%	13.03%
	OPT	28.75%	2.7%	12.62%
	Paraphrase	28.39%	3%	12.26%
	Back-Translation	28.03%	2.36%	11.9%
	Token-LM	27.16%	**1.67%**	11.03%
	Char	25.5%	7.02%	9.37%
	Spel	29.05%	2.16%	12.92%
	Lexical	26.93%	5.02%	10.8%
Low-resource	**GAN-LM**	**10.15%**	1.95%	**9.27%**
	GPT	8.48%	3.61%	7.6%
	OPT	8.17%	1.9%	7.29%
	Paraphrase	5.93%	2.42%	5.05%
	Back-Translation	7.27%	1.59%	6.39%
	Token-LM	5.26%	3.72%	4.38%
	Char	4.19%	**1.42%**	3.31%
	Spel	7.68%	4.03%	6.8%
	Lexical	6.09%	3.3%	5.21%
	Baseline - Low	0.88%	1.54%	-
	Baseline - Half	16.13%	1.16%	-
	Baseline - Normal	24.97%	2.27%	-

Semantic Textual Similarity in *STS-B*

Scenarios	Type	SRC	CI	Change
Half-train set	**GAN-LM**	**78.02%**	0.96%	**4.44%**
	GPT	76.94%	0.83%	3.36%
	OPT	76.97%	1.65%	3.39%
	Paraphrase	77.07%	2.01%	3.49%
	Back-Translation	77.1%	2.4%	3.52%
	Token-LM	76.11%	**0.57%**	2.53%
	Char	75.43%	0.86%	1.85%
	Spel	76.61%	2.13%	3.03%
	Lexical	76.74%	1.39%	3.16%
Low-resource	**GAN-LM**	**61.66%**	**1.46%**	**23.44%**
	GPT	58.11%	6.38%	19.89%
	OPT	59.17%	3.95%	20.95%
	Paraphrase	57.9%	3.1%	19.68%
	Back-Translation	58.02%	6.72%	19.8%
	Token-LM	56.66%	2.59%	18.44%
	Char	53.32%	1.6%	15.1%
	Spel	54.52%	5.07%	16.3%
	Lexical	57.77%	5.17%	19.55%
	Baseline - Low	38.22%	10.61%	-
	Baseline - Half	73.58%	4.08%	-
	Baseline - Normal	78.49%	0.28%	-

this environment and can be a top pick which has 7.17% F1 improvement with half-train set against full train set without augmentation.

5.4 Results in Semantic Textual Similarity

Table 2 right side covers the results on the *STS-B* dataset. In this application, we need various and semantically closed augmented data to improve the result. The size of training data is 1K from 115 baseline in low-resource and 8K from 2K baseline in half-train set scenario. In low-resource, we can achieve great improvements, especially with contextual-level and GAN-LM approaches. In half-train set, the improvement is smaller than the one in low-resource setting but we can see consistent improvements by including synthetic data. Again, contextual-level augmentations outperforms non-contextual-level and GAN-LM yields the best performance for semantic textual similarity (STS) task which gives a closed performance as the result from full train set without augmentation.

5.5 Results in Multilingual Semantic Textual Similarity

Lastly, we extend our work to multilingual task. Table 3 shows the results on the *mSTS* dataset. In this task, we target diverse and semantically consistent augmented samples in multilingual to enhance the performance. The amount of train set is 800 from 200 baseline in low-resource, and 4K from 2K baseline in normal scenario. In low-resource, we can confirm that all augmentations improve the overall performance, especially with GAN-LM. Compared to low-resource, the improvement in normal is lower but still, GAN-LM mostly gives the best results, except for EN-AR. This is because GAN-LM is mostly trained on Indo-European languages (i.e. EN, DE, NL, FR, ES, IT) which enhances the generation ability for these languages. Interestingly, GAN-LM works well in EN-TR since the performance without augmentation in this pair is very low and it has a large gap to be improved by augmentation, especially with GAN-LM which saves original structure with similarity thresholds and does token-level tweaking with affordable diversity learned from train set. We can find that back-translation works the best in EN-AR because it directly uses the well-defined neural translation models for augmentation which finally decreases the unsuitable assigned languages (e.g. code-switching) suffered by other augmentations. To complement our approach, we combine GAN-LM with back-translation, called GAN-LM-Back, to enhance the performance. In this method, we generate the synthetic data for AR-AR and EN-EN using back-translation and other monolingual pairs

Table 3: SRC values in *mSTS* with 95% CI. Here, we focus on the contextual-level augmentations which promise the superior performances in STS task. Denotations are identical as Table 1.

Scenarios	Type	EN-AR	ES-EN	EN-DE	EN-TR	FR-EN	IT-EN	NL-EN	CI	Change
Normal	**GAN-LM -Back**	46.18%	**55.92%**	**59.23%**	**43.72%**	60.93%	**57.32%**	53.9%	2.64%	**2.38%**
	GAN-LM	44.44%	53.6%	59.2%	42.62%	**61.48%**	55.31%	**53.96%**	**2.62%**	1.43%
	mGPT	45.24%	50.86%	59.2%	42.52%	60.51%	53.07%	53.86%	2.71%	0.67%
	Paraphrase	45.21%	48.69%	58.06%	40.9%	60.67%	54.12%	53.32%	2.92%	0.06%
	Back-Translation	**46.36%**	50.62%	57.26%	41.82%	58.64%	53.48%	52.98%	2.72%	0.08%
Low-resource	**GAN-LM**	**31.75%**	**37.05%**	**44.71%**	**24.21%**	**43.12%**	**39.96%**	**43.96%**	**3.06%**	**5.43%**
	mGPT	30.29%	34.33%	38.11%	19.64%	34.9%	33.37%	39.19%	4.83%	0.44%
	Paraphrase	28.67%	35.93%	37.76%	22.04%	35.4%	32.63%	35.24%	3.59%	0.13%
	Back-Translation	31.01%	34.44%	36.67%	21.94%	36.28%	31.7%	37.15%	4.49%	0.35%
	Baseline - Low	29.95%	33.13%	36.04%	18.23%	37.26%	34.68%	37.46%	3.85%	-
	Baseline - Normal	45.08%	50.52%	56.9%	40.94%	60.89%	53.16%	53.08%	2.47%	-

using GAN-LM to fine-tune the mean pooling of mBERT. Using GAN-LM-Back, we achieve the overall enhancements. Thus, we can understand that GAN-LM and its extension, GAN-LM-Back, are the best approaches for mutlilingual STS task.

5.6 Analysis of Synthetic Data

In this section, we analyze the synthetic data from each augmentation method. Table 4 shows examples of synthetic data in *TREC* dataset. Lexical-based finds the synonym of the word, spelling and character-based tweak the words, and token-LM-based changes the lowercase word and auxiliary verb. Both back-translation and paraphrase restate a text with different orders and words while both OPT and GPT adds a new context after original statement. Interestingly, GAN-LM focuses on changing question word which is the main factor to increase the performance. It also finds a semantically similar word. Lastly, we can see that GAN-LM-GPT is the combination between GAN-LM and GPT. From our findings, DA improves model robustness to unseen noisy inputs in downstream tasks. Augmentations containing grammatical mistakes, speech recognition errors, semantically similar terms help the model generalize better. With GAN-LM, we preserve both the semantics and the structure of the input text, while providing diverse augmentations. More examples of augmented data are covered in Appendix.

5.7 Ablation Study

In Table 5, we example the ablation study of GAN-LM where 0.3-0.7 range as the similarity thresholds and BART as pre-trained LM are the best choices in *STS-B*. The similarity thresholds control the analogy and diversity of synthetic data where 0.3-0.7 range was the top choice in *STS-B* to balance these

Table 4: Examples of generated augmentations. Bold texts in each cell mean the changed parts.

Type	Example
Original	Why do heavier objects travel downhill faster ?
Lexical	Why do heavier object travel downhill **quicker**?
Spelling	**Whay** do heavier objects travel downhill faster?
Character	Why do heavier **osbjects tralvel downhzill** faster?
Token-LM	**WHY does** heavier objects travel downhill faster ?
Back-Translation	Why **are the** heavier objects **moving down faster?**
Paraphrase	Why do heavier objects **go faster downhill?**
OPT	Why do heavier objects travel downhill faster ? **Because they're heavier**
GPT	Why do heavier objects travel downhill faster ? **Or slow down to 2 km h**
GAN-LM	**HOW** do heavier objects travel **down** faster ?
GAN-LM-GPT	**HOW** do heavier objects travel **down** faster ? **Or slow down to 2 km h**

two terms for achieving the best performance. Similar patterns are observed in other downstream tasks, except for *mSTS* where mBART and 0.5-0.9 range are selected. Additional architectural ablation study is shown in Appendix.

In addition, we investigate the effect of size of augmented data in Table 6. We consider the validation set (or cross-validation for dataset without validation set) to determine the optimal size of augmented data and find that there is a specific point when the validation performance becomes stable. Our findings indicate that performances (SRC - Test in Table 6) are stabilized after this certain point, implying that the generated synthetic data offers sufficient diversity to improve the model's generalization capabilities. The size of augmentation in other tasks are determined by same approach.

6 Conclusions

In this work, we investigate the effect of different DAs to improve the performance on various tasks. We study both techniques found in the literature as well as the proposed GAN-LM in different scenarios: We subsample training sets to study model per-

Table 5: GAN-LM study in *STS-B* with half-train set.

Type	SRC
GAN-LM with BART (0.3 - 0.7)	**78.02%**
GAN-LM with BART (0.1 - 0.5)	75.57%
GAN-LM with BART (0.5 - 0.9)	77.49%
GAN-LM with BERT (0.3 - 0.7)	71.33%
GAN-LM with XLNet (0.3 - 0.7)	74.21%

Table 6: Investigation on the size of train set with GAN-LM. Validation and Test describe each set in *STS-B*.

Low-resource in *STS-B*			
Type	**Size of Train set**	**SRC - Validation**	**SRC - Test**
GAN-LM	690	65.56%	56.81%
	920	68.93%	60.16%
	1150 (same as Table 2)	**71%**	**61.66%**
	1380	70.89%	61.61%

formance under low-resource conditions and use half or full training set to understand under different conditions. In most experiments, GAN-LM clearly gives the better results than non-contextual and contextual-level augmentations. In addition to apply GAN-LM solely, we combine it with GPT and back-translation to supplement the performance.

7 Limitations

There are three predictable limitations in the developed GAN-LM. First, the convergence of training process in GAN-LM should be investigated carefully. Different datasets have different distribution of data and characteristics which can affect the GAN-LM's convergence and we need a few iterations of training to confirm the suitable epochs for each task. Second, there can be a machine bias since each model is trained on machine generated synthetic data. Therefore, searching the suitable pre-trained model is important to be considered at the beginning. Last, while we did a thorough evaluation of GAN-LM on four downstream tasks, it is still a general-purpose approach and its effectiveness on specific tasks or domains may vary. Thus, further research is required to fully understand its capabilities and limitations in different contexts.

Supplementary Materials Availability Statement: Source code is included in supplementary materials. Notes on reproducibility (e.g. computational budget and used hyperparameters) are included in Appendix. Additional ablation study and augmented examples are covered in Appendix. Links for considered datasets and models are shown in Appendix.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples.

Antreas Antoniou, Amos Storkey, and Harrison Edwards. 2017. Data augmentation generative adversarial networks.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Claude Coulombe. 2018. Text data augmentation made simple by leveraging nlp cloud apis. *arXiv preprint arXiv:1812.04718*.

Danilo Croce, Giuseppe Castellucci, and Roberto Basili. 2020. GAN-BERT: Generative adversarial learning for robust text classification with a bunch of labeled examples. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2114–2119, Online. Association for Computational Linguistics.

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.

Juri Ganitkevitch and Chris Callison-Burch. 2014. The multilingual paraphrase database. In *The 9th edition* *of the Language Resources and Evaluation Conference*, Reykjavik, Iceland. European Language Resources Association.

O. Yu. Golovneva and Charith S. Peris. 2020. Generative adversarial networks for annotated data augmentation in data sparse nlu. In *ICON*.

Ishaan Gulrajani, Faruk Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. 2017. Improved training of wasserstein gans. In *NIPS*.

Michael A. Hedderich, Lukas Lange, Heike Adel, Jannik Strötgen, and Dietrich Klakow. 2021. A survey on recent approaches for natural language processing in low-resource scenarios. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2545–2568, Online. Association for Computational Linguistics.

Helsinki-NLP. 2023. Github - helsinki-nlp/opus-mt: Open neural machine translation models and web services.

Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Chin-Yew Lin, and Deepak Ravichandran. 2001. Toward semantics-based answer pinpointing. In *Proceedings of the First International Conference on Human Language Technology Research*.

Ehsan Kamalloo, Mehdi Rezagholizadeh, Peyman Passban, and Ali Ghodsi. 2021. Not far away, not so close: Sample efficient nearest neighbour data augmentation via MiniMax. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3522–3533, Online. Association for Computational Linguistics.

Akbar Karimi, Leonardo Rossi, and Andrea Prati. 2021. AEDA: An easier data augmentation technique for text classification. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2748–2754, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Matt J Kusner and José Miguel Hernández-Lobato. 2016. Gans for sequences of discrete elements with the gumbel-softmax distribution. *arXiv preprint arXiv:1611.04051*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *ArXiv*, abs/1910.13461.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Lajanugen Logeswaran, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Jacob Devlin, and Honglak Lee. 2019. Zero-shot entity linking by reading entity descriptions. *arXiv preprint arXiv:1906.07348*.

Petro Marek, Vishal Ishwar Naik, Vincent Auvray, and Anuj Goyal. 2021. Oodgan: Generative adversarial network for out-of-domain data generation. *ArXiv*, abs/2104.02484.

Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *ICLR*.

George A. Miller, Richard Beckwith, Christiane D. Fellbaum, Derek Gross, and Katherine J. Miller. 1990. Introduction to wordnet: An on-line lexical database. *International Journal of Lexicography*, 3:235–244.

Weili Nie, Nina Narodytska, and Ankit Patel. 2018. Relgan: Relational generative adversarial networks for text generation. In *International conference on learning representations*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. 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. Association for Computational Linguistics.

Danish Pruthi, Bhuwan Dhingra, and Zachary Chase Lipton. 2019. Combating adversarial misspellings with robust word recognition. In *ACL*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks.

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. *ArXiv*, abs/1511.06709.

Haoyue Shi, Karen Livescu, and Kevin Gimpel. 2021. Substructure substitution: Structured data augmentation for NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3494–3508, Online. Association for Computational Linguistics.

Ziyi Shou, Yuxin Jiang, and Fangzhen Lin. 2022. AMR-DA: Data augmentation by Abstract Meaning Representation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3082–3098, Dublin, Ireland. Association for Computational Linguistics.

Sandeep Subramanian, Sai Rajeswar, Francis Dutil, Christopher Pal, and Aaron Courville. 2017. Adversarial generation of natural language. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 241–251.

Zhixing Tan, Xiangwen Zhang, Shuo Wang, and Yang Liu. 2021. Msp: Multi-stage prompting for making pre-trained language models better translators.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual translation with extensible multilingual pretraining and finetuning.

Nandan Thakur, Nils Reimers, Johannes Daxenberger, and Iryna Gurevych. 2021. Augmented SBERT: Data augmentation method for improving bi-encoders for pairwise sentence scoring tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 296–310, Online. Association for Computational Linguistics.

Brian Thompson and Matt Post. 2020a. Automatic machine translation evaluation in many languages via zero-shot paraphrasing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online. Association for Computational Linguistics.

Brian Thompson and Matt Post. 2020b. Paraphrase generation as zero-shot multilingual translation: Disentangling semantic similarity from lexical and syntactic diversity. In *Proceedings of the Fifth Conference on Machine Translation (Volume 1: Research Papers)*, Online. Association for Computational Linguistics.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models.

William Yang Wang and Diyi Yang. 2015. That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using #petpeeve tweets. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2557–2563, Lisbon, Portugal. Association for Computational Linguistics.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks.

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2019. Scalable zero-shot entity linking with dense entity retrieval. *arXiv preprint arXiv:1911.03814*.

Yiben Yang, Chaitanya Malaviya, Jared Fernandez, Swabha Swayamdipta, Ronan Le Bras, Ji-Ping Wang, Chandra Bhagavatula, Yejin Choi, and Doug Downey. 2020. Generative data augmentation for commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1008–1025, Online. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31.

Huangzhao Zhang, Hao Zhou, Ning Miao, and Lei Li. 2019a. Generating fluent adversarial examples for natural languages. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5564–5569, Florence, Italy. Association for Computational Linguistics.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019b. PAWS: Paraphrase Adversaries from Word Scrambling. In *Proc. of NAACL*.