Adversarial Augmentation Policy Search for Domain and Cross-Lingual Generalization in Reading Comprehension

Adyasha Maharana **Mohit Bansal**
UNC Chapel Hill
{adyasha, mbansal}@cs.unc.edu

Abstract

Reading comprehension models often overfit to nuances of training datasets and fail at adversarial evaluation. Training with adversarially augmented dataset improves robustness against those adversarial attacks but hurts generalization of the models. In this work, we present several effective adversaries and automated data augmentation policy search methods with the goal of making reading comprehension models more robust to adversarial evaluation, but also improving generalization to the source domain as well as new domains and languages. We first propose three new methods for generating QA adversaries, that introduce multiple points of confusion within the context, show dependence on insertion location of the distractor, and reveal the compounding effect of mixing adversarial strategies with syntactic and semantic paraphrasing methods. Next, we find that augmenting the training datasets with uniformly sampled adversaries improves robustness to the adversarial attacks but leads to decline in performance on the original unaugmented dataset. We address this issue via RL and more efficient Bayesian policy search methods for automatically learning the best augmentation policy combinations of the transformation probability for each adversary in a large search space. Using these learned policies, we show that adversarial training can lead to significant improvements in in-domain, out-of-domain, and cross-lingual (German, Russian, Turkish) generalization.[1]

1 Introduction

There has been growing interest in understanding NLP systems and exposing their vulnerabilities through maliciously designed inputs (Iyyer et al., 2018; Belinkov and Bisk, 2018; Nie et al., 2019; Gurevych and Miyao, 2018). Adversarial examples are generated using search (Alzantot et al., 2018), heuristics (Jia and Liang, 2017) or gradient (Ebrahimi et al., 2018) based techniques to fool the model into giving the wrong outputs. Often, the model is further trained on those adversarial examples to make it robust to similar attacks. In the domain of reading comprehension (RC), adversaries are QA samples with distractor sentences that have significant overlap with the question and are randomly inserted into the context. By having a fixed template for creating the distractors and training on them, the model identifies learnable biases and overfits to the template instead of being robust to the attack itself (Jia and Liang, 2017). Hence, we first build on Wang and Bansal (2018)'s work of adding randomness to the template and significantly expand the pool of distractor candidates by introducing multiple points of confusion within the context, adding dependence on insertion location of the distractor, and further combining distractors with syntactic and semantic paraphrases to create combinatorially adversarial examples that stress-test the model's language understanding capabilities. These adversaries inflict up to 45% drop in performance of RC models built on top of large pretrained models like RoBERTa (Liu et al., 2019).

Next, to improve robustness to the aforementioned adversaries, we finetune the RC model with a combined augmented dataset containing an equal number of samples from all of the adversarial transformations. While it improves robustness by a significant margin, it leads to decline in performance on the original unaugmented dataset. Hence, instead of uniformly sampling from the various adversarial transformations, we propose to perform a search for the best adversarial policy combinations that improve robustness against the adversarial attacks and also preserve/improve accuracy on the original dataset via data augmentation. However, it

[1] We will publicly release all our code, adversarial policy data, and models on our webpage.

Findings of the Association for Computational Linguistics: EMNLP 2020, pages 3723–3738

is slow, expensive and inductive-biased to manually tune the transformation probability for each adversary and repeat the process for each target dataset, and so we present RL and Bayesian search methods to learn this policy combination automatically.

For this, we create a large augmentation search space of up to 10^6, with four adversarial methods, two paraphrasing methods and a discrete binning of probability space for each method (see Figure 1). Cubuk et al. (2019) showed via AutoAugment that a RNN controller can be trained using reinforcement learning to find the best policy in a large search space. However, AutoAugment is computationally expensive and relies on the assumption that the policy searched using rewards from a smaller model and reduced dataset will generalize to bigger models. Alternatively, the augmentation methods can be modelled with a surrogate function, such as Gaussian processes (Rasmussen, 2003), and subjected to Bayesian optimization (Snoek et al., 2012), drastically reducing the number of training iterations required for achieving similar results (available as a software package for computer vision).[2] Hence, we extend these ideas to NLP and perform a systematic comparison between AutoAugment and our more efficient BayesAugment.

Finally, there has been limited previous work exploring the role of adversarial data augmentation to improve generalization of RC models to out-of-domain and cross-lingual data. Hence, we also perform automated policy search of adversarial transformation combinations for enhancing generalization from English Wikipedia to datasets in other domains (news, web) and languages (Russian, German, Turkish). Policy search methods like BayesAugment can be readily adapted for low-resource scenarios where one only has access to a small development set that the model can use as a black-box evaluation function (for rewards, but full training or gradient access on that data is unavailable). We show that augmentation policies for the source domain learned using target domain performance as reward, improves the model's generalization to the target domain with only the use of a small development set from that domain. Similarly, we use adversarial examples in a pivot language (in our case, English) to improve performance on other languages' RC datasets using rewards from a small development set from that language.

Our contributions can be summarized as follows:

- We first propose novel adversaries for reading comprehension that cause up to 45% drop in large pretrained models' performance. Augmenting the training datasets with uniformly sampled adversaries improves robustness to the adversarial attacks but leads to decline in performance on the original unaugmented dataset.
- We next demonstrate that optimal adversarial policy combinations of transformation probabilities (for augmentation and generalization) can be automatically learned using policy search methods. Our experiments show that efficient Bayesian optimization achieves similar results as AutoAugment with a fraction of the resources.
- By training on the augmented data generated via the learned policies, we not only improve adversarial robustness of the models but also show significant gains i.e., up to 2.07%, 5.0%, and 2.21% improvement for in-domain, out-of-domain, and cross-lingual evaluation respectively.

  Overall, the goal of our paper is to make reading comprehension models robust to adversarial attacks as well as out-of-distribution data in cross-domain and cross-lingual scenarios.

2 Related Work

Adversarial Methods in NLP: Following the introduction of adversarial evaluation for RC models by Jia and Liang (2017); Wang and Bansal (2018), several methods have been developed for probing the sensitivity and stability of NLP models (Nie et al., 2019; Glockner et al., 2018). Zhao et al. (2018) employ GANS to generate semantically meaningful adversaries. Ren et al. (2019) and Alzantot et al. (2018) use a synonym-substitution strategy while Ebrahimi et al. (2018) create gradient-based perturbations. Iyyer et al. (2018) construct a syntactic paraphrasing network to introduce syntactic variance in adversaries.

Augmentation and Generalization: Goodfellow et al. (2015) and Miyato et al. (2018) use adversarial training to demonstrate improvement in image recognition. Xie et al. (2020) improve the adversarial training scheme with auxiliary batch normalization modules. Back-translation (Yu et al., 2018), pre-training with other QA datasets (Devlin et al., 2019; Lewis et al., 2019; Talmor and Berant, 2019) and virtual adversarial training (Miyato et al., 2017; Yang et al., 2019) are shown to be effective augmentation techniques for RC datasets. Cao et al. (2020) propose a conditional adversarial

[2] https://pypi.org/project/deepaugment/

Adversary Method	Description	Original Question/Sentence and Corresponding Distractor
AddSentDiverse	(Jia and Liang, 2017; Wang and Bansal, 2018)	Q: In what country is Normandy located? D: *D-Day* is located in the country of *Sri Lanka.*
AddKSentDiverse	Multiple AddSentDiverse distractors are inserted randomly in the context.	Q: Which county is developing its business center? D1: The county of Switzerland is developing its art periphery. D2: The county of Switzerland is developing its *home center*.
AddAnswerPosition	Answer span is preserved in this distractor. It is most misleading when inserted before the original answer.	Q: What is the steam engine's thermodynamic basis? A: The Rankine cycle is the fundamental thermodynamic underpinning of the steam engine. D: Rankine cycle is the *air* engine's thermodynamic basis.
InvalidateAnswer	AddSentDiverse and additional elimination of the original answer.	Q: Where has the official home of the Scottish Parliament been since 2004? D: Since *October 2002*, the *unofficial abroad* of the *Welsh Assembly* has been a *old Welsh Assembly Houses*, in the *Golden Gate Bridge* area of *Glasgow*.
PerturbAnswer	Content words (except named entities) are algorithmically replaced with synonyms and evaluated for consistency using language model.	A: The UK refused to sign the Social Charter and was exempt from the legislation covering Social Charter issues unless it agreed to be bound by the legislation. P: The UK *repudiated* to *signature* the Social Charter and was exempt from the legislation *encompassing* Social Charter issues unless it *consented* to be *related* by the legislation.
PerturbQuestion	Syntacting paraphrasing network is used to generate the source question with a different syntax.	Q: In what country is Normandy located? P: *Where* does Normany *exist*?

Table 1: Demonstration of the various adversary functions used in our experiments (Q=Question, D=Distractor, A=Answer, P=Paraphrase). Words that have been modified using adversarial methods are italicized in the distractor.

self-training method to reduce domain distribution discrepancy. Lee et al. (2019); Wang et al. (2019) use a discriminator to enforce domain-invariant representation learning (Fisch et al., 2019); Chen et al. (2018) and Zhang et al. (2017) learn language-invariant representations for cross-lingual tasks. We show that heuristics-based adversaries can be used for augmentation as well as generalization.

Policy Search: Cubuk et al. (2019) present the AutoAugment algorithm which uses reinforcement learning to find the best augmentation policies in a large search space, and then follow-up with RandAugment (Cubuk et al., 2020) which reduces the task to simple grid-search. Niu and Bansal (2019) use AutoAugment to discover perturbation policies for dialogue generation. Ho et al. (2019) use population-based augmentation (PBA) techniques (Jaderberg et al., 2017) and significantly reduce the compute time required by AutoAugment. We are the first to adapt RandAugment style techniques for NLP via our BayesAugment method. RandAugment enforces uniform transformation probability on all augmentation methods and collapses the augmentation policy search space to two global parameters. BayesAugment eliminates the need to choose between adversarial methods and optimizes only for their transformation probabilities (see Sec. 3.2).

3 Adversary Policy Design

As shown by Jia and Liang (2017), QA models are susceptible to random, semantically meaningless and minor changes in the data distribution. We extend this work and propose adversaries that exploit the model's sensitivity to insert location of distractor, number of distractors, combinatorial adversaries etc. After exposing the model's weaknesses, we strengthen them by training on these adversaries and show that the model's robustness to adversarial attacks significantly increases due to it. Finally, in Sec. 4, we automatically learn the right combination of transformation probability for each adversary in response to a target improvement using policy search methods.

3.1 Adversary Transformations

We present two types of adversaries, namely positive perturbations and negative perturbations (or attacks) (Figure 1). Positive perturbations are adversaries generated using methods that have been traditionally used for data augmentation in NLP i.e., semantic and syntactic transformations. Negative perturbations are distractor sentences based on the classic AddSent model (Jia and Liang, 2017) that exploits the RC model's shallow language understanding to mislead it to incorrect answers. We use

the method outlined by Wang and Bansal (2018) for **AddSentDiverse** to generate a distractor sentence (see Table 1) and insert it randomly within the context of a QA sample.

We introduce more variance to adversaries with **AddKSentDiverse**, wherein multiple distractor sentences are generated using AddSentDiverse and are inserted at independently sampled random positions within the context. For **AddAnswerPosition**, the original answer span is retained within the distractor sentence and the model is penalized for incorrect answer span location. We remove the sentence containing the answer span from the context and introduce a distractor sentence to create **InvalidateAnswer** adversarial samples which are no longer answerable. **PerturbAnswer** adversaries are created by following the `Perturb` subroutine (Alzantot et al., 2018) and generating semantic paraphrases of the sentence containing the answer span. We use the syntactic paraphrase network (Iyyer et al., 2018) to create **PerturbQuestion** adversarial samples by replacing the original question with its paraphrase.

Finally, we combine negative and positive perturbations to create adversaries which double-down on the model's language understanding. It always leads to a larger drop in performance when tested on the RC models trained on original unaugmented datasets. See Appendix for more details.

3.2 Adversarial Policy & Search Space

Reading comprehension models are often trained with adversarial samples in order to improve robustness to the corresponding adversarial attack. We seek to find the best combination of adversaries for data augmentation that also preserves/improves accuracy on source domain and improves generalization to a different domain or language.

AutoAugment: Following previous work in AutoAugment policy search (Cubuk et al., 2019; Niu and Bansal, 2019), we define a sub-policy to be a set of adversarial transformations which are applied to a QA sample to generate an adversarial sample. We show that adversaries are most effective when positive and negative perturbations are applied together (Table 2). Hence, to prepare one sub-policy, we select one of the four negative perturbations (or none), combine it with one of the two positive perturbations (or none) and assign the combination a transformation probability (see Figure 1). The probability space $[0, 1]$ is discretized into 6 equally spaced bins. This leads to a search space of $5 * 3 * 6 = 90$ for a single sub-policy. Next, we define a complete adversarial policy as a set of n sub-policies with a search space of 90^n. For each input QA sample, one of the sub-policies is randomly sampled and applied (with a probability equal to the transformation probability) to generate the adversarial sample. Thus, each original QA sample ends up with one corresponding adversarial sample or none.

BayesAugment: We adopt a simplified formulation of the policy for our BayesAugment method, following Ho et al. (2019) and RandAugment (Cubuk et al., 2020). Sampling of positive and negative adversaries is eliminated and transformation probabilities of all possible combinations of adversaries are optimized over a continuous range $[0, 1]$.[3] Consequently, one of these combinations is randomly sampled for each input QA sample to generate adversaries. Empirically, the dominant adversary in a policy is the attack with highest transformation probability (see policies in Table 8 in Appendix). Due to the probabilistic nature of the policy, it is possible for the model to not add any adversarial sample at all, but the probability of this happening is relatively low.

4 Automatic Policy Search

Next, we need to perform search over the large space of augmentation policies in order to find the best policy for a desired outcome. Performing naive search (random or grid) or manually tuning the transformation probabilities is slow, expensive and largely impractical due to resource constraints. Hence, we compare two different approaches for learning the best augmentation policy in fewer searches: AutoAugment and BayesAugment. We follow the optimization procedure as demonstrated in Figure 1. For $t = 1, 2, ...,$ do:

- Sample the next policy p_t (*sample*)
- Transform training data with p_t and generate augmented data (*apply, transform*)
- Train the downstream task model with augmented data (*train*)

[3] RandAugment collapses a large parameter space by enforcing uniform probability on all transformations and optimizing for: (i) global distortion parameter, (ii) number of transformations applied to each image. It uses hyperparameter optimization and shows results with naive grid search due to small search space. RandAugment is not directly applicable to our setting because there is no notion of global distortion for text. Hence, we borrow the idea of treating augmentation policy parameters as hyperparameters but use Bayesian optimization for search.

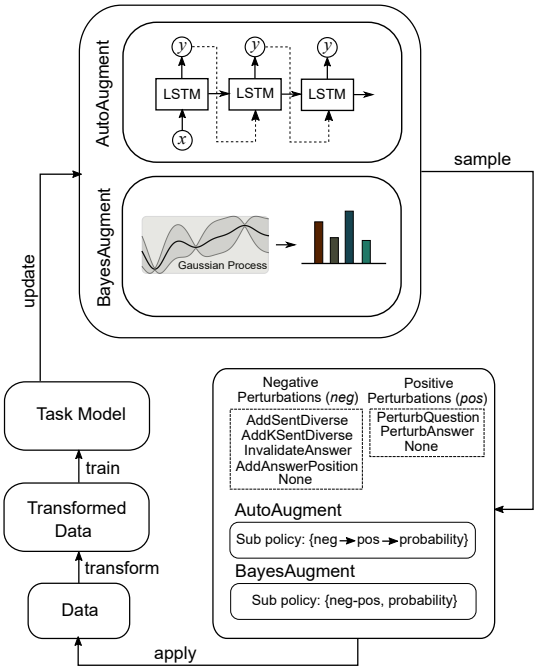

Figure 1: Flow chart of training loop for AutoAugment controller and Bayesian optimizer. See Sec. 4.

- Obtain score on validation dataset as reward r_t
- Update Gaussian Process or RNN Controller with r_t (*update*)

4.1 AutoAugment

Our AutoAugment model (see Figure 1) consists of a recurrent neural network-based controller and a downstream task model. The controller has n output blocks for n sub-policies; each output block generates distributions for the three components of sub-policies i.e., *neg*, *pos* and *probability*. The adversarial policy is generated by sampling from these distributions and applied on input dataset to create adversarial samples, which are added to the original dataset to create an augmented dataset. The downstream model is trained on the augmented dataset till convergence and evaluated on a given metric, which is then fed back to the controller as a reward (see the *update* flow in figure). We use REINFORCE (Sutton et al., 1999; Williams, 1992) to train the controller.

4.2 BayesAugment

Typically, it takes thousands of steps to train an AutoAugment controller using reinforcement learning which prohibits the use of large pretrained models as task model in the training loop. For example, the controllers in Cubuk et al. (2019) were trained for 15,000 samples or more. To circumvent this computational issue, we frame our adversarial policy search as a hyperparameter optimization problem and use Bayesian methods to perform the search. Bayesian optimization techniques use a surrogate model to approximate the objective function f and an acquisition function to sample points from areas where improvement over current result is most likely. The prior belief about f is updated with samples drawn from f in order to get a better estimate of the posterior that approximates f. Bayesian methods attempt to find global maximum in the minimum number of steps.

4.3 Rewards

The F1 score of downstream task model on development set is used as reward during policy search. To discover augmentation policies which are geared towards improving generalization of RC model, we calculate the F1 score of task model (trained on source domain) on out-of-domain or cross-lingual development datasets, and feed it as the reward to the optimizer.

4.4 Datasets

We use SQuAD v2.0 (Rajpurkar et al., 2018) and NewsQA (Trischler et al., 2017) for adversarial evaluation and in-domain policy-search experiments. Futher, we measure generalization from SQuAD v2.0 to NewsQA and TriviaQA (Joshi et al., 2017), and from SQuAD v1.1 (Rajpurkar et al., 2016) to German dataset from MLQA (Lewis et al., 2020) and Russian, Turkish datasets from XQuAD (Artetxe et al., 2020).[4] See Appendix for more details on datasets and training.

4.5 Reading Comprehension Models

We use RoBERTa$_{\text{BASE}}$ as the primary RC model for all our experiments. For fair baseline evaluation on out-of-domain and cross-lingual datasets, we also use the development set of the target task to select the best checkpoint. Search algorithms like AutoAugment require a downstream model that can be trained and evaluated fast, in order to reduce training time. So, we use distilRoBERTa$_{\text{BASE}}$ (Sanh et al., 2019) for AutoAugment training loops. BayesAugment is trained for fewer iterations than AutoAugment and hence, allows us to use RoBERTa$_{\text{BASE}}$ model directly in the training loop. See Appendix for more details and baseline performances of these models.

[4]The choice of cross-lingual datasets in our experiments is based on availability of x-en translation and span alignment models for the Translate-Test method (Asai et al., 2018)

Adversary Method	SQuAD	NewsQA
Baseline (No Adversaries)	81.17	58.40
AddSentDiverse	65.50	51.47
AddKSentDiverse (K=2)	45.31	48.31
AddAnswerPosition	68.91	49.20
InvalidateAnswer	77.75	24.03
PerturbQuestion	43.67	36.76
PerturbAnswer	71.97	59.08
Effect of Multiple Distractors		
AddSentDiverse	65.50	51.47
Add2SentDiverse	45.31	48.31
Add3SentDiverse	43.49	44.81
Combinatorial effect		
AddSentDiverse	65.50	51.47
+ PerturbAnswer	50.71	51.43
AddKSentDiverse	45.31	48.31
+ PerturbQuestion	31.56	29.56
Effect of Insert Location of AddAnswerPosition		
Random	68.91	49.20
Prepend	66.52	48.01
Append	67.84	48.76

Table 2: Adversarial evaluation of baseline RoBERTa$_{\text{BASE}}$ trained on SQuAD v2.0 and NewsQA. Compare to corresponding rows in Table 3 to observe difference in performance after adversarial training. Results (F1 score) are shown on dev set.

4.6 Evaluation Metrics

We use the official SQuAD evaluation script for evaluation of robustness to adversarial attacks and performance on in-domain and out-of-domain datasets.[5] For cross-lingual evaluation, we use the modified Translate-Test method as outlined in Lewis et al. (2020); Asai et al. (2018). QA samples in languages other than English are first translated to English and sent as input to RoBERTa$_{\text{BASE}}$ finetuned on SQuAD v1.1. The predicted answer spans within English context are then mapped back to the context in original language using alignment scores from the translation model. We use the top-ranked German→English and Russian→English models in WMT19 shared news translation task, and train a Turkish→English model using a similar architecture, to generate translations and alignment scores (Ng et al., 2019).[6]

5 Results

First, in Sec. 5.1, we perform adversarial evaluation of baseline RC models for various categories of adversaries. Next, in Sec. 5.2, we train the RC models with an augmented dataset that contains equal ratios of adversarial samples and show that it improves robustness to adversarial attacks but hurts performance of the model on original unaugmented dataset. Finally, in Sec. 5.3, we present results from AutoAugment and BayesAugment policy search and the in-domain, out-of-domain and cross-lingual performance of RC models trained using augmentation data generated from the learned policies with corresponding target rewards.

[5]Statistical significance is computed with 100K samples using bootstrap (Noreen, 1989; Tibshirani and Efron, 1993).

[6]https://github.com/pytorch/fairseq

Adversary Method	SQuAD	NewsQA
AddSentDiverse	68.00	61.13
AddKSentDiverse (K=2)	79.44	62.31
AddAnswerPosition	80.16	56.90
InvalidateAnswer	91.41	67.57
PerturbQuestion	60.91	44.99
PerturbAnswer	76.42	60.74
Original Dev (No Adversaries)	78.83	58.08

Table 3: Adversarial evaluation after training RoBERTa$_{\text{BASE}}$ with the original dataset augmented with equally sampled adversarial data. Compare to corresponding rows in Table 2 to observe difference in performance after adversarial training. Results (F1 score) are shown on dev set.

5.1 Adversarial Evaluation

Table 2 shows results from adversarial evaluation of RoBERTa$_{\text{BASE}}$ finetuned with SQuAD v2.0 and NewsQA respectively. All adversarial methods lead to a significant drop in performance for the finetuned models i.e., between 4-45% for both datasets. The decrease in performance is maximum when there are multiple distractors in the context (Add3SentDiverse) or perturbations are combined with one another (AddSentDiverse + PerturbAnswer). These results show that, in spite of being equipped with a broader understanding of language from pretraining, the finetuned RC models are shallow and over-stabilized to textual patterns like n-gram overlap. Further, the models aren't robust to semantic and syntactic variations in text.

Additionally, we performed manual evaluation of 96 randomly selected adversarial samples (16 each from attacks listed in Table 1) and found that a human annotator picked the right answer for 85.6% of the questions.

5.2 Manual Adversarial Training

Next, in order to remediate the drop in performance observed in Table 2 and improve robustness to adversaries, the RC models are further finetuned for 2 epochs with an adversarially augmented training set. The augmented training set contains each

Search Method	*In-domain* SQuAD	NewsQA	SQuAD → NewsQA	TriviaQA
Validation				
Base	81.17 / 77.54	58.40 / 47.04	48.36 / 36.06	41.60 / 34.86
UniS	78.83 / 74.68	58.08 / 46.79	48.24 / 36.03	42.04 / 35.11
Auto	81.63 / 78.06	**62.17 / 49.41**	**50.57 / 38.56**	42.41 / 35.41
Bayes	**81.71 / 78.12**	58.62 / 47.21	49.73 / 38.38	**43.96 / 36.67**
Test				
Base	80.64 / 77.19	57.02 / 45.29	44.95 / 34.68	36.01 / 29.23
UniS	78.42 / 75.87	57.21 / 45.36	46.30 / 35.94	37.83 / 30.52
Auto	**81.06 / 77.79**	**59.09 / 45.49**	46.82 / 35.75	37.88 / 30.60
Bayes	80.88 / 77.57	57.63 / 45.32	**48.95 / 37.44**	**40.99 / 33.68**

Table 4: Baseline results (first row) and evaluation after finetuning baseline models with the adversarial policies derived from AutoAugment and BayesAugment for in-domain improvements and out-of-domain generalization from Wikipedia (SQuAD) to news (NewsQA) and web (TriviaQA) domains. Results (F1 / Exact Match) are shown on validation and test sets. (Base=Baseline, UniS=Uniform Sampling, Auto=AutoAugment, Bayes=BayesAugment)

Search Method	*Cross-lingual generalization from English SQuAD →* MLQA (de)	XQuAD (ru)	XQuAD (tr)
Validation			
Baseline	58.58 / 36.41	67.89 / 44.62	42.95 / 25.09
UniformS	58.97 / 36.68	68.11 / 44.84	43.12 / 25.26
BayesAug	**59.40 / 37.11**	**68.73 / 45.34**	**44.09 / 25.73**
Test			
Baseline	57.56 / 36.01	60.81 / 33.47	40.49 / 23.14
UniformS	58.27 / 36.45	61.87 / 34.31	41.04 / 23.78
BayesAug	**59.02 / 38.01**	**63.03 / 34.85**	**41.95 / 24.17**

Table 5: Cross-lingual QA: Translate-Test (Lewis et al., 2020) evaluation after finetuning the baseline with adversarial policies derived from BayesAugment for generalization to German (de), Russian (ru) and Turkish (tr) RC datasets. Results (F1 / Exact Match) are shown on validation and test sets.

QA sample from the original training set and a corresponding adversarial QA sample by randomly sampling from one of the adversary methods. Table 3 shows results from adversarial evaluation after adversarial training. Adding perturbed data during training considerably improves robustness of the models to adversarial attacks. For instance, RoBERTa$_{\text{BASE}}$ performs with 79.44 F1 score on SQuAD AddKSentDiverse samples (second row, Table 3), as compared to 45.31 F1 score without adversarial training (third row, Table 2). Similarly, RoBERTa$_{\text{BASE}}$ performs with 44.99 F1 score on NewsQA PerturbQuestion samples (fifth row, Table 3), as compared to a baseline score of 36.76 F1 score (sixth row, Table 2). However, this manner of adversarial training also leads to drop in performance on the original unaugmented development set, e.g., RoBERTa$_{\text{BASE}}$ performs with 78.83 and 58.08 F1 scores on the SQuAD and NewsQA development sets respectively, which is 2.34 and 0.32 points lesser than the baseline (first row, Table 2).

5.3 Augmentation Policy Search for Domain and Language Generalization

Following the conclusion from Sec. 5.2 that uniform sampling of adversaries is not the optimal approach for model performance on original unaugmented dataset, we perform automated policy search over a large search space using BayesAugment and AutoAugment for in-domain as well as cross-domain/lingual improvements (as discussed in Sec. 4). For AutoAugment, we choose the number of sub-policies in a policy to be $n = 3$ as a trade-off between search space dimension and optimum results. We search for the best transformation policies for the source domain that lead to improvement of the model in 3 areas: 1. in-domain performance 2. generalization to other domains and 3. generalization to other languages. These results are presented in Tables 4 and 5, adversarial evaluation of the best BayesAugment models is presented in Table 6, and the learned policies are shown in the Appendix.

In-domain evaluation: The best AutoAugment augmentation policies for improving in-domain performance of RoBERTa$_{\text{BASE}}$ on the development sets result in 0.46% and 3.77% improvement in F1 score over baseline for SQuAD v2.0 and NewsQA respectively (see Table 4). Similarly, we observe 0.54% (p=0.021) and 0.22% (p=0.013) absolute improvement in F1 Score for SQuAD and NewsQA respectively by using BayesAugment policies. This trend is reflected in results on the test set as well. AutoAugment policies result in most improvement i.e., 0.42% (p=0.014) and 2.07% (p=0.007) for SQuAD and NewsQA respectively. Additionally, both policy search methods outperform finetuning with a dataset of uniformly sampled adversaries (see row 2 in Table 4).

Out-of-domain evaluation: To evaluate generalization of the RC model from Wikipedia to news articles and web, we train RoBERTa$_{\text{BASE}}$ on SQuAD and evaluate on NewsQA and TriviaQA respectively. The baseline row in Table 4 presents results of RoBERTa$_{\text{BASE}}$ trained on original unaugmented SQuAD and evaluated on NewsQA and TriviaQA. Next, we reiterate results from Table 3 and show that finetuning with uniformly sampled dataset (see UniS in Table 4) of adversaries results in drop in performance on the validation sets of

SQuAD and NewsQA. By training on adversarially augmented SQuAD with AutoAugment policy, we see 2.21% and 0.81% improvements on the development sets of NewsQA (SQuAD→NewsQA) and TriviaQA (SQuAD→TriviaQA) respectively. Similarly, BayesAugment provides 1.37% and 2.36% improvements over baseline for development sets of TriviaQA and NewsQA, proving as a competitive and less computationally intensive substitute to AutoAugment. BayesAugment outperforms AutoAugment at out-of-domain generalization by providing 4.0%($p<0.001$) and 4.98% jump on test sets for NewsQA and TriviaQA respectively, as compared to 1.87% improvements with AutoAugment.

Our experiments suggest that AutoAugment finds better policies than BayesAugment for in-domain evaluation. We hypothesize that this might be attributed to a difference in search space between the two policy search methods. AutoAugment is restricted to sampling at most 3 sub-policies while BayesAugment has to simultaneously optimize the transformation probability for ten or more different augmentation methods. A diverse mix of adversaries from the latter is shown to be more beneficial for out-of-domain generalization but results in minor improvements for in-domain performance. Moving ahead, due to better performance for out-of-domain evaluation and more efficient trade-off with computation, we only use BayesAugment for our cross-lingual experiments.

Cross-lingual evaluation: Table 5 shows results of $\text{RoBERTa}_{\text{BASE}}$ finetuned with adversarially augmented SQuAD v1.1[7] and evaluated on RC datasets in non-English languages. The baseline row presents results from $\text{RoBERTa}_{\text{BASE}}$ trained on original unaugmented SQuAD and evaluated on German MLQA(de), Russian XQuAD(ru) and Turkish XQuAD(tr) datasets; F1 scores on the development sets are 58.58, 67.89 and 42.95 respectively. These scores depend on quality of the translation model as well as the RC model. We observe significant improvements on the development as well as test sets by finetuning baseline RC model with adversarial data from English SQuAD. Uniformly sampled adversarial dataset results in 0.71% (p=0.063), 1.06% (p=0.037), and 0.55% (p=0.18) improvement for test sets of MLQA(de), XQuAD(ru) and XQuAD(tr), respectively. BayesAugment policies outperform uniform sampling and result in 1.47% (p=0.004), 2.21% (p=0.007) and 1.46% (p=0.021) improvement for test sets of MLQA(de), XQuAD(ru) and XQuAD(tr), respectively.

Adversarial evaluation: We show results from the adversarial evaluation of $\text{RoBERTa}_{\text{BASE}}$ models finetuned with adversarially augmented SQuAD using policies learned from BayesAugment in Table 6. We use the best models for out-of-domain and cross-lingual generalization as shown in Tables 4 and 5, and evaluate their performance on the adversaries discussed in Section 5.1. Results show that the policies learnt from BayesAugment significantly improve resilience to the proposed adversarial attacks in addition to improving performance on the target datasets. The performance on adversaries varies with the transformation probability of the respective adversaries in the learned policies. For example, the transformation probability of *PerturbQuestion* adversaries is 0.83 and 0.0 for SQuAD→TriviaQA and SQuaD→NewsQA models respectively (see Table 8). Consequently, the former has a higher performance on *PerturbQuestion* adversaries.

6 Analysis and Discussion

Having established the efficacy of automated policy search for adversarial training, we further probe the robustness of adversarially trained models to unseen adversaries. We also analyze the convergence of BayesAugment for augmentation policy search and contrast its requirement of computational resources with that of AutoAugment. See Appendix for more analysis on domain independence of adversarial robustness and augmentation data size.

Robustness to Unseen Adversaries: We train $\text{RoBERTa}_{\text{BASE}}$ on SQuAD v2.0 augmented with the AddSentDiverse counterpart of each QA sample and evaluate it on other adversarial attacks, to analyze robustness of the model to unseen adversaries. As seen from the results in Table 7, training with AddSentDiverse leads to large improvements on AddKSentDiverse and small improvements on PerturbQuestion and PerturbAnswer i.e., 31.21% (45.31 vs. 76.52), 1.56% (43.67 vs. 45.23) and 5.31% (71.97 vs. 77.28) respectively, showing that the model becomes robust to multiple distractors within the same context and it also gains some resilience to paraphrasing operations. Conversely, we see a drop in performance on InvalidateAnswer, showing that it is easier for the model to be dis-

[7] *InvalidateAnswer* adversaries are not used for generalization from SQuADv1.1 because it does not contain the NoAnswer style samples introduced in SQuADv2.0.

Adversary Method	*Out-of-domain generalization* TriviaQA	NewsQA	*Cross-lingual generalization* MLQA (de)	XQuAD (ru)	XQuAD (tr)
AddSentDiverse	67.17 / 65.60	66.26 / 64.59	63.68 / 61.09	65.21 / 64.04	65.17 / 63.83
AddKSentDiverse (K=2)	78.48 / 76.32	77.13 / 75.80	76.91 / 74.45	77.76 / 75.20	77.93 / 75.37
AddAnswerPosition	80.05 / 77.41	79.46 / 76.31	78.62 / 75.59	80.24 / 77.38	79.51 / 76.28
InvalidateAnswer	88.23 / 85.56	90.18 / 78.25	-	-	-
PerturbQuestion	60.39 / 58.02	54.65 / 51.48	58.14 / 56.33	60.15 / 57.92	59.71 / 56.27
PerturbAnswer	77.12 / 75.38	76.30 / 74.12	77.28 / 75.82	74.31 / 72.88	74.72 / 73.16

Table 6: Adversarial evaluation after finetuning the baseline with adversarial policies derived from BayesAugment for generalization from SQuAD2.0 to TriviaQA, NewsQA, and SQuAD1.1 to German (de), Russian (ru) and Turkish (tr) RC datasets. Results (F1 / Exact Match) are shown on validation sets. Compare to corresponding rows in Table 3 to observe difference in performance between models finetuned with uniformly sampled dataset vs. dataset derived from learned policies.

Adversary Attack	Trained on SQuAD	Trained on SQuAD+AddSentDiverse
AddKSentDiverse	45.31	76.52
InvalidateAnswer	77.75	70.91
PerturbQuestion	43.67	45.23
PerturbAnswer	71.97	77.28

Table 7: Robustness of $\text{RoBERTa}_{\text{BASE}}$ trained on a subset of adversaries to unseen adversaries. Results (F1 score) are shown on SQuAD dev set.

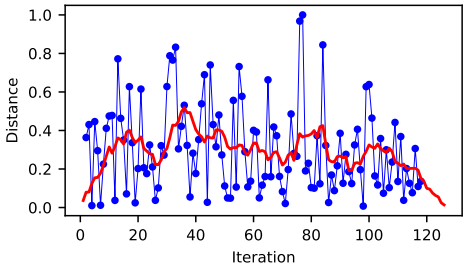

Figure 2: Demonstration of variation in distance between neighboring samples picked by Bayesian optimizer with increasing training iterations. The red line represents moving average of distances.

tracted by adversaries when the original answer is removed from context.

Bayesian Convergence: In comparison to the thousands of training loops or more for AutoAugment, we run BayesAugment for only 100 training loops with 20 restarts. To show that BayesAugment converges within the given period, we plot the distance between transformation probabilities chosen by the Bayesian optimizer for the AddSentDiverse-PerturbQuestion augmentation method. As shown in Figure 2, the distance between the samples decreases with progression in training, showing that the optimizer becomes more confident about the narrow range of probability which should be sampled for maximum performance on validation set.

Analysis of Resources for AutoAugment vs BayesAugment: With lesser number of training loops, BayesAugment uses only 10% of the GPU resources required for AutoAugment. Our AutoAugment experiments have taken more than 1000 iterations and upto 5-6 days for convergence, requiring many additional days for hyperparameter tuning. In contrast, our BayesAugment experiment ran for 36-48 hours on 2 1080Ti GPUs and achieved comparable performance with 100 iterations or less. If large pretrained models are replaced with smaller distilled models in future work, BayesAugment will provide even more gains in time/computation.

7 Conclusion

We show that adversarial training can be leveraged to improve robustness of reading comprehension models to adversarial attacks and also to improve performance on source domain and generalization to out-of-domain and cross-lingual data. We present BayesAugment for policy search, which achieves results similar to the computationally-intensive AutoAugment method but with a fraction of computational resources. By combining policy search with rewards from the corresponding target development sets' performance, we show that models trained on SQuAD can be generalized to NewsQA and German, Russian, Turkish cross-lingual datasets without any training data from the target domain or language.

Acknowledgments

We thank the reviewers for their useful feedback. This work was supported by DARPA MCS Grant #N66001-19-2-4031, DARPA KAIROS Grant #FA8750-19-2-1004, ONR Grant #N00014-18-1-2871, and awards from Google, Facebook, and Amazon (plus Amazon and Google GPU cloud credits). The views are those of the authors and not of the funding agency.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896, Brussels, Belgium. Association for Computational Linguistics.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In *ACL*.

Akari Asai, Akiko Eriguchi, Kazuma Hashimoto, and Yoshimasa Tsuruoka. 2018. Multilingual extractive reading comprehension by runtime machine translation. *arXiv preprint arXiv:1809.03275*.

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Yu Cao, Meng Fang, Baosheng Yu, and Joey Tianyi Zhou. 2020. Unsupervised domain adaptation on reading comprehension. In *AAAI*.

Xilun Chen, Yu Sun, Ben Athiwaratkun, Claire Cardie, and Kilian Weinberger. 2018. Adversarial deep averaging networks for cross-lingual sentiment classification. *Transactions of the Association for Computational Linguistics*, 6:557–570.

Ekin D. Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V. Le. 2019. Autoaugment: Learning augmentation strategies from data. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. 2020. RandAugment: Practical data augmentation with no separate search. In *CVPR Workshops*.

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.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. HotFlip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36, Melbourne, Australia. Association for Computational Linguistics.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 1–13, Hong Kong, China. Association for Computational Linguistics.

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking NLI systems with sentences that require simple lexical inferences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 650–655, Melbourne, Australia. Association for Computational Linguistics.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *ICLR*.

Iryna Gurevych and Yusuke Miyao. 2018. Proceedings of the 56th annual meeting of the association for computational linguistics (volume 1: Long papers). In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Daniel Ho, Eric Liang, Xi Chen, Ion Stoica, and Pieter Abbeel. 2019. Population based augmentation: Efficient learning of augmentation policy schedules. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2731–2741. PMLR.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885, New Orleans, Louisiana. Association for Computational Linguistics.

Max Jaderberg, Valentin Dalibard, Simon Osindero, Wojciech M Czarnecki, Jeff Donahue, Ali Razavi, Oriol Vinyals, Tim Green, Iain Dunning, Karen Simonyan, et al. 2017. Population based training of neural networks. *DeepMind tech report. arXiv preprint arXiv:1711.09846*.

Robin Jia and Percy Liang. 2017. 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. Association for Computational Linguistics.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611.

Seanie Lee, Donggyu Kim, and Jangwon Park. 2019. Domain-agnostic question-answering with adversarial training. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 196–202, Hong Kong, China. Association for Computational Linguistics.

Patrick Lewis, Ludovic Denoyer, and Sebastian Riedel. 2019. Unsupervised question answering by cloze translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4896–4910, Florence, Italy. Association for Computational Linguistics.

Patrick Lewis, Barlas Oğuz, Ruty Rinott, Sebastian Riedel, and Holger Schwenk. 2020. MLQA: Evaluating cross-lingual extractive question answering. In *ACL*.

Chia-Wei Liu, Ryan Lowe, Iulian V Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *EMNLP*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Takeru Miyato, Andrew M. Dai, and Ian J. Goodfellow. 2017. Adversarial training methods for semi-supervised text classification. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. 2018. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1979–1993.

Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. 2019. Facebook FAIR's WMT19 News Translation Task Submission. In *WMT*.

Yixin Nie, Yicheng Wang, and Mohit Bansal. 2019. Analyzing compositionality-sensitivity of NLI models. In *AAAI*, pages 6867–6874.

Tong Niu and Mohit Bansal. 2018. Adversarial over-sensitivity and over-stability strategies for dialogue models. In *CoNLL*.

Tong Niu and Mohit Bansal. 2019. Automatically learning data augmentation policies for dialogue tasks. 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 1317–1323, Hong Kong, China. Association for Computational Linguistics.

Eric W Noreen. 1989. *Computer-intensive methods for testing hypotheses*. Wiley New York.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *OpenAI Technical Report*.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

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

Carl Edward Rasmussen. 2003. Gaussian processes in machine learning. In *Summer School on Machine Learning*, pages 63–71. Springer.

Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. 2019. Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1085–1097, Florence, Italy. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In *5th Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS 2019*.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. 2012. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 4444–4451. AAAI Press.

Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. 2010. Gaussian process optimization in the bandit setting: No regret and experimental design. In *ICML*.

Richard S. Sutton, David A. McAllester, Satinder P. Singh, and Yishay Mansour. 1999. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems 12, [NIPS Conference, Denver, Colorado, USA, November 29 - December 4, 1999]*, pages 1057–1063. The MIT Press.

Alon Talmor and Jonathan Berant. 2019. MultiQA: An empirical investigation of generalization and transfer in reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4911–4921, Florence, Italy. Association for Computational Linguistics.

Robert J Tibshirani and Bradley Efron. 1993. An introduction to the bootstrap. *Monographs on statistics and applied probability*, 57:1–436.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200, Vancouver, Canada. Association for Computational Linguistics.

Huazheng Wang, Zhe Gan, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, and Hongning Wang. 2019. Adversarial domain adaptation for machine reading comprehension. In *EMNLP*.

Yicheng Wang and Mohit Bansal. 2018. Robust machine comprehension models via adversarial training. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 575–581, New Orleans, Louisiana. Association for Computational Linguistics.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8:229–256.

Cihang Xie, Mingxing Tan, Boqing Gong, Jiang Wang, Alan Yuille, and Quoc V Le. 2020. Adversarial examples improve image recognition. In *CVPR*.

Ziqing Yang, Yiming Cui, Wanxiang Che, Ting Liu, Shijin Wang, and Guoping Hu. 2019. Improving machine reading comprehension via adversarial training. *arXiv preprint arXiv:1911.03614*.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. 2018. QANet: Combining Local Convolution with Global Self-Attention for Reading Comprehension. In *ICLR*. OpenReview.net.

Meng Zhang, Yang Liu, Huanbo Luan, and Maosong Sun. 2017. Adversarial training for unsupervised bilingual lexicon induction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1959–1970.

Zhengli Zhao, Dheeru Dua, and Sameer Singh. 2018. Generating natural adversarial examples. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Appendix

A Adversary Transformations

We present two types of adversaries, namely positive perturbations and negative perturbations (or attacks). Positive perturbations are adversaries generated using methods that have been traditionally used for data augmentation in NLP i.e., semantic and syntactic transformations. Negative perturbations are adversaries based on the classic AddSent model (Jia and Liang, 2017) that exploit the RC model's shallow language understanding to mislead it to incorrect answers.

AddSentDiverse: We use the method outlined by Wang and Bansal (2018) for AddSentDiverse to generate a distractor sentence and insert it randomly within the context of a QA sample. In addition to WordNet, we use ConceptNet (Speer et al., 2017) for a wider choice of antonyms during generation of adversary. QA pairs that do not have an answer within the given context are also augmented with AddSentDiverse adversaries.

AddKSentDiverse: The AddSentDiverse method is used to generate multiple distractor sentences for a given context. Each of the distractor sentences is then inserted at independently sampled random positions within the context. The distractors may or may not be similar to each other. Introducing multiple points of confusion is a more effective technique for misleading the model and reduces the scope of learnable biases during adversarial training by adding variance.

AddAnswerPosition: The original answer span is retained and placed within a distractor sentence generated using a combination of AddSentDiverse and random perturbations to maximize semantic mismatch. We modify the evaluation script to compare exact answer span locations in addition to the answer phrase and fully penalize incorrect locations. For practical purposes, if the model predicts the answer span within adversarial sentence as output, it does not make a difference. However, it brings into question the interpretability of such models. This distractor is most effective when placed right before the original answer sentence, showing dependence on insert location of distractor.

InvalidateAnswer: The sentence containing the original answer is removed from the context. Instead, a distractor sentence generated using AddSentDiverse is introduced to the context. This

method is used to augment the adversarial *NoAnswer*-style samples in SQuAD v2.0.

PerturbAnswer (Semantic Paraphrasing): Following Alzantot et al. (2018), we perform semantic paraphrasing of the sentence containing the answer span. Instead of using genetic algorithm, we adapt their `Perturb` subroutine to generate paraphrases in the following steps:

1. Select word locations for perturbations, which includes locations within any content phrase that does not appear within the answer span. Here, content phrases are verbs, adverbs and adjectives.
2. For location k_i in the set of word locations $\{k\}$, compute 20 nearest neighbors of the word at given location using GloVe embeddings, create a candidate sentence by perturbing the word location with each of the substitute words and rank perturbed sentences using a language model.
3. Select the perturbed sentence with highest rank and perform Step 2 for the next location k_{i+1} using the perturbed sentence.

We use the OpenAI-GPT model (Radford et al., 2018) to evaluate paraphrases.

PerturbQuestion (Syntactic Paraphrasing): We use the syntactic paraphrase network introduced by Iyyer et al. (2018) to generate syntactic adversaries. Sentences from the context of QA samples tend to be long and have complicated syntax. The corresponding syntactic paraphrases generated by the paraphrasing network usually miss out on half of the source sentence. Therefore, we choose to perform paraphrasing on the questions. We generate 10 paraphrases for each question and rank them based on cosine similarity, computed between the mean of word embeddings (Pennington et al., 2014) of source sentence and generated paraphrases (Niu and Bansal, 2018; Liu et al., 2016).

Finally, we combine negative perturbations with positive perturbations to create adversaries which double-down on the model's language understanding capabilities. It always leads to a larger drop in performance when tested on the reading comprehension models trained on original unaugmented datasets.

Semantic Difference Check: To make sure that the distractor sentences are sufficiently different from the original sentence, we perform a semantic difference check in two steps:

1. Extract content phrases from original sentence. Content phrase is any common NER phrase or one of the four: noun, verb, adverb, adjective.
2. There should be at least 2 content phrases in the original text that aren't found in the distractor.

We examined 100 randomly sampled original-distractor sentence pairs and found that our semantic difference check works for 96% of the cases.

B BayesAugment

We use Gaussian Process (GP) (Rasmussen, 2003) as surrogate function and Upper Confidence Bound (UCB) (Srinivas et al., 2010) as the acquisition function. GP is a non-parametric model that is fully characterized by a mean function $\mu_0 : \chi \mapsto \mathbb{R}$ and a positive-definite kernel or covariance function $k : \chi \times \chi \mapsto \mathbb{R}$. Let $x_1, x_2, ...x_n$ denote any finite collections of n points, where each x_i represents a choice of sampling probabilities for each of the augmentation methods and $f_i = f(x_i)$ is the (unknown) function value evaluated at x_i. Let $y_1, y_2, ...y_n$ be the corresponding noisy observations (the validation performance at the end of training). In the context of GP Regression (GPR), $f = f_1, ...f_n$ are assumed to be jointly Gaussian. Then, the noisy observations $y = y_1, ...y_n$ are normally distributed around f as $y|f \sim \mathcal{N}(f, \sigma^2 I)$. The Gaussian Process upper confidence bound (GP-UCB) algorithm measures the optimistic performance upper bound of the sampling probabilities.

C Datasets

SQuAD v2.0 (Rajpurkar et al., 2018) is a crowd-sourced dataset consisting of 100,000 questions from SQuAD v1.1 (Rajpurkar et al., 2016) and an additional 50,000 questions that do not have answers within the given context. We split the official development set into 2 randomly sampled sets of validation and test for our experiments.

NewsQA is also a crowd-sourced extractive RC dataset based on 10,000 news articles from CNN, containing both answerable and unanswerable questions. (Trischler et al., 2017) To accommodate very long contexts from NewsQA in models like Bert (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), we sample two instances from the set of overlapping instances for the final training data.

TriviaQA (Joshi et al., 2017) questions were crawled from the web and have two variants. One variant includes Wikipedia articles as contexts; we use the other variant which involves web snippets and documents from Bing search engine as contexts. The development and test sets are large

AutoAugment Policies	
SQuAD → SQuAD	(AddS, None, 0.2) → (IA, None, 0.4) → (AddA, None, 0.2)
SQuAD → NewsQA	(None, PA, 0.4) → (None, PA, 0.6) → (AddS, PA, 0.4)
SQuAD → TriviaQA	(AddS, None, 0.9) → (AddS, PA, 0.7) → (AddKS, PQ, 0.9)
NewsQA → NewsQA	(AddA, PA, 0.2) → (AddKS, None, 0.2) → (AddA, PA, 0.4)
BayesAugment Policies	
SQuAD → SQuAD	(AddS, 0.29), (AddA, 0.0), (AddA-PA, 0.0), (AddA-PQ, 0.0), (AddKS, 0.0), (AddKS-PA,0.0) (AddKS-PQ, 0.0), (AddS-PA, 0.0), (AddS-PQ, 0.0), (PA, 0.61), (PQ, 0.0), (IA, 1.0)
SQuAD → NewsQA	(AddS, 1.0), (AddA, 0.0), (AddA-PA, 1.0), (AddA-PQ, 0.0), (AddKS, 0.0), (AddKS-PA, 0.0) (AddKS-PQ, 0.0), (AddS-PA, 1.0), (AddS-PQ, 0.0), (PA, 0.48), (PQ, 0.0), (IA, 0.0)
SQuAD → TriviaQA	(AddS, 1.0), (AddA, 1.0), (AddA-PA, 0.21), (AddA-PQ, 0.18), (AddKS, 0.86), (AddKS-PA, 0.37) (AddKS-PQ, 0.25), (AddS-PA, 0.12), (AddS-PQ, 0.49), (PA, 0.91), (PQ, 0.83), (IA, 0.26)
SQuAD → MLQA(de)	(AddS, 0.042), (AddA-PA, 0.174), (AddA-PQ, 0.565), (AddKS, 0.173), (AddKS-PA, 0.567) (AddA, 0.514), (AddS-PA, 0.869), (AddS-PQ, 0.720), (PA, 0.903), (PQ, 0.278), (AddKS-PQ, 0.219)
SQuAD → XQuAD(ru)	(AddS, 0.147), (AddA-PA, 0.174), (AddA-PQ, 0.79), (AddKS, 0.55), (AddKS-PA, 0.97) (AddA, 0.77), (AddS-PA, 0.02), (AddS-PQ, 0.59), (PA, 0.11), (PQ, 0.95), (AddKS-PQ, 0.725)
SQuAD → XQuAD(tr)	(AddS, 0.091), (AddA-PA, 0.463), (AddA-PQ, 0.64), (AddKS, 0.32), (AddKS-PA, 0.86) (AddA, 0.34), (AddS-PA, 0.37), (AddS-PQ, 0.43), (PA, 0.27), (PQ, 0.81), (AddKS-PQ, 0.493)
NewsQA → NewsQA	(AddS, 1.0), (AddA, 1.0), (AddA-PA, 1.0), (AddA-PQ, 0.0), (AddKS, 0.0), (AddKS-PA, 1.0) (AddKS-PQ, 0.156), (AddS-PA, 0.0), (AddS-PQ, 0.720), (PA, 0.0), (PQ, 0.0), (IA, 1.0)

Table 8: Best Policies suggested by BayesAugment and AutoAugment methods for different scenarios; AddS = AddSentDiverse, AddKS = AddKSentDiverse, AddA = AddAnswerPosition, IA = InvalidateAnswer, PA = PerturbAnswer, PQ = PerturbQuestion.

Model	SQuADv1.1	SQuADv2.0	NewsQA
RoBERTa	89.73 / 82.38	81.17 / 77.54	58.40 / 47.04
DistilRoBERTa	84.57 / 75.81	73.29 / 69.47	54.21 / 42.76

Table 9: Comparison of performance (F1 Score / Exact Match) of different models on SQuAD v1.1, SQuaD v2.0 and NewsQA datasets. $\text{RoBERTa}_{\text{BASE}}$ is the baseline model; $\text{DistilRoBERTa}_{\text{BASE}}$ is the task model used during AutoAugment policy search.

Hyperparameter	**SQuAD v1.1**	**SQuAD v2.0**	**NewsQA**
Learning Rate	3e-5	1.5e-5	1.6e-5
Batch Size	24	16	24
Warmup Ratio	0.06	0.06	0.08
No. of Epochs	2	5	5
Weight Decay	0.01	0.01	0.01

Table 10: Best hyperparameters for training $\text{RoBERTa}_{\text{BASE}}$ on SQuAD v2.0 and NewsQA.

with more than 60K samples in each. For faster BayesAugment and AutoAugment iterations, we randomly select 10K samples from the development set to generate rewards.

MLQA (Lewis et al., 2020) is the multilingual extension to SQuAD v1.1 consisting of evaluation (development and test) data only. We use German (de) MLQA in our experiments.

XQuAD is a multilingual version of SQuAD (Artetxe et al., 2020) containing only test sets. We use Russian (ru) and Turkish (tr) XQuAD which contain nearly 1100 QA samples that are further split equally and randomly into development and test sets.

D Training Details

Reading Comprehension Models: We use $\text{RoBERTa}_{\text{BASE}}$ as the primary RC model for all our experiments. Search algorithms like AutoAugment require a downstream model that can be trained and evaluated fast, in order to reduce training time. So, we use $\text{distilRoBERTa}_{\text{BASE}}$ (Sanh et al., 2019) for AutoAugment training loops, which has 40% lesser parameters than $\text{RoBERTa}_{\text{BASE}}$. It should be noted that the distilRoBERTa model used in our experiments is trained on SQuAD without distillation. BayesAugment is trained for fewer iterations than AutoAugment and hence, allows us to use $\text{RoBERTa}_{\text{BASE}}$ model directly in the training loop.

Model Hyperparameters: We trained

NewsQA Adversary	SQuAD	SQuAD → NewsQA
AddSentDiverse	42.39 / 32.79	49.54 / 38.02
PerturbAnswer	39.95 / 27.60	45.52 / 32.49
AddSentDiv-PertrbAns	35.08 / 26.33	43.63 / 32.76

Table 11: Comparison of robustness between RoBERTa$_{\text{BASE}}$ finetuned on original unaugmented SQuAD and our best SQuAD → NewsQA generalized model. Results (F1 score/Exact Match) are shown on dev set.

Augmentation Ratio	**NewsQA**
RoBERTa	48.36 / 36.06
+ 1x augmentation	49.73 / 38.38
+ 2x augmentation	49.84 / 37.97
+ 3x augmentation	49.62 / 38.01

Table 12: Effect of augmentation ratio for generalization from SQuAD→NewsQA. Results (F1 score/Exact Match) are shown on NewsQA dev set.

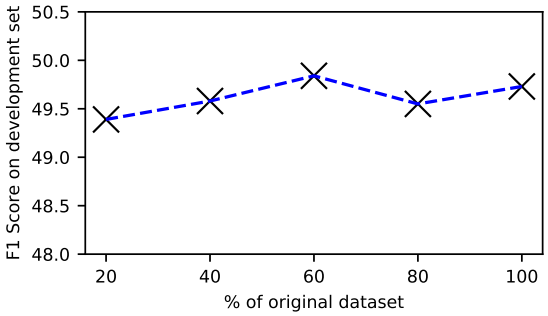

Figure 3: Performance of SQuAD → NewsQA model on NewsQA dev set (F1 score) with increasing size of finetuning dataset.

RoBERTa$_{\text{BASE}}$ for 5 epochs on SQuAD and NewsQA respectively and selected the best-performing checkpoint as baseline. We perform a hyperparameter search for both datasets using Bayesian optimization search (Snoek et al., 2012). The RNN controller in AutoAugment training loop consists of a single LSTM cell with a single hidden layer and hidden layer dimension of 100. The generated policy consists of 3 sub-policies; each sub-policy is structured as discussed in main text. BayesAugment is trained for 100 iterations with 20 restarts. During AutoAugment and BayesAugment training loops, RoBERTa$_{\text{BASE}}$ or distilRoBERTa$_{\text{BASE}}$ (which has already been trained on unaugmented SQuAD) is further finetuned on the adversarially augmented dataset for 2 epochs with a warmup ratio of 0.2 and learning rate decay (lr=1e-5) thereafter. After the policy search, further hyperparameter optimization is performed for best results from fine-tuning. We do not perform this last step of hyperparameter tuning on cross-lingual data to avoid the risk of overfitting the small datasets. For generalization from SQuAD v1.1 to cross-lingual datasets, we do not consider the adversary InvalidateAnswer because *NoAnswer* samples do not exist for these datasets.

E Analysis

In this section, we show the impact of adversarial augmentation ratio in training dataset and the size of training dataset on the generalization of RC model to out-of-domain data. Next, we show more experiments on robustness to unseen adversaries. Finally, we analyze the domain-independence of adversarial robustness by training on adversarially augmented SQuAD and testing on adversarial NewsQA samples.

Effect of Augmentation Ratio: To assess the importance of adversarial augmentation in the dataset, we experimented with different ratios i.e., 1x, 2x and 3x, of augmented samples to the original dataset, for generalization from SQuAD to NewsQA using the augmentation policy learnt by BayesAugment. The performance of SQuAD→NewsQA models on NewsQA validation set were 49.73, 49.84 and 49.62 for 1x, 2x and 3x augmentations respectively, showing slight improvement for twice the number of augmentations. However, the performance starts decreasing at 3x augmentations, showing that too many adversaries in the training data starts hurting generalization.

Effect of Augmented Dataset Size: We experimented with 20%, 40%, 60%, 80% and 100% of the original dataset to generate augmented dataset using the BayesAugment policy for generalization of RoBERTa$_{\text{BASE}}$ trained on SQuAD to NewsQA and observed little variance in performance with increasing data, as seen from Figure 3. The augmentation ratio in these datasets is 1:1. We hypothesize that the model is saturated early on during training, within the first tens of thousands of adversarially augmented samples. Exposing the model to more SQuAD samples gives little boost to performance on NewsQA thereafter.

Robustness to Unseen Adversaries: We train RoBERTa$_{\text{BASE}}$ on SQuAD which has been augmented with an adversarial dataset of the same size as SQuAD and contains equal number of samples

Adversary Attack	Trained on SQuAD	Trained on SQ+ASD/PQ/PA
AddSentDiverse+PerturbAnswer	50.71	84.37
AddKSentDiverse+PerturbQuestion	31.56	78.91
AddAnswerPosition	68.91	80.87
AddKSentDiverse	45.31	76.14
InvalidateAnswer	77.75	71.62

Table 13: Robustness of RoBERTa$_{\text{BASE}}$ trained on a subset of adversaries to unseen adversaries. Results (F1 score) are shown on SQuAD dev set (ASD=AddSentDiverse, PQ=PerturbQuestion, PA=PerturbAnswer, SQ=SQuAD).

Hyperparameter	Range
Learning Rate	$[1e^{-5}, 2e^{-5}]$
Batch Size	$\{8, 16, 24, 32\}$
Warmup Ratio	$[0.01, 0.5]$
Weight Decay	$[0.01, 0.1]$

Table 14: Bayesian Optimization Ranges for Finetuning RoBERTA with AutoAugment and Bayesaugment policies (32 iterations with 8 restarts).

from AddSentDiverse, PerturbQuestion and PerturbAnswer. In Table 13, We see that the model is significantly more robust to combinatorial adversaries like AddSentDiverse+PerturbAnswer when trained on the adversaries AddSentDiverse and PerturbAnswer individually. We also see a decline in performance on InvalidateAnswer.

Domain-Independence of Robustness to Adversarial Attacks: We have shown that a reading comprehension model trained on SQuAD can be generalized to NewsQA by finetuning the model with adversarially transformed samples from SQuAD dataset. It is expected that this model will be robust to similar attacks on SQuAD. To assess if this robustness generalizes to NewsQA as well, we evaluate our best SQuAD→NewsQA model on adversarially transformed NewsQA samples from the development set. The SQuAD column in Table 11 shows results from evaluation of RoBERTa$_{\text{BASE}}$ finetuned with original unaugmented SQuAD, on adversarially transformed NewsQA samples. Interestingly, the generalized model (rightmost column) is 5-8% more robust to adversarial NewsQA without being trained on any NewsQA samples, showing that robustness to adversarial attacks in source domain easily generalizes to a different domain.