Problem / Task / Question
Idea / Model / Solution
Results / Findings
Critique

Data Boost: Text Data Augmentation Through Reinforcement Learning Guided Conditional Generation

Ruibo Liu[1], Guangxuan Xu[2], Chenyan Jia[3], Weicheng Ma[4], Lili Wang[5], and Soroush Vosoughi[6]

[1,2,4,5,6]Department of Computer Science, Dartmouth College
[3]Moody College of Communication, University of Texas at Austin
[1,4,5]{ruibo.liu.gr,weicheng.ma.gr,lili.wang.gr}@dartmouth.edu
[2]guangxuan.xu.ug@dartmouth.edu
[3]chenyanjia@utexas.edu
[6]soroush.vosoughi@dartmouth.edu

Abstract

Data augmentation is proven to be effective in many NLU tasks, especially for those suffering from data scarcity. In this paper, we present a powerful and easy to deploy text augmentation framework, **Data Boost**, which augments data through reinforcement learning guided conditional generation. We evaluate Data Boost on three diverse text classification tasks under five different classifier architectures. The result shows that Data Boost can boost the performance of classifiers especially in low-resource data scenarios. For instance, Data Boost improves F1 for the three tasks by 8.7% on average when given only 10% of the whole data for training. We also compare Data Boost with six prior text augmentation methods. Through human evaluations (N=178), we confirm that Data Boost augmentation has comparable quality as the original data with respect to readability and class consistency.

1 Introduction

Data augmentation is a widely-used technique in classification tasks. In the field of computer vision (CV), data is augmented by flipping, cropping, tilting, and altering RGB channels of the original images (Krizhevsky et al., 2012; Chatfield et al., 2014; Szegedy et al., 2015); however, similar intuitive and simple strategies do not obtain equal success in NLP tasks. Existing methods tend to produce augmentation with low readability or unsatisfying semantic consistency (Yang et al., 2020).

Table 1 shows some output samples of popular text augmentation methods. Naive methods imitate pixel manipulation in CV, augmenting sentences by adding spelling errors (Xie et al., 2017), or randomly deleting and swapping tokens (Wei and Zou, 2019). The output of such augmentation methods are often illegible since the word order is disrupted (e.g., *"is The baby very!"*); even worse, crucial feature words (e.g., the word *lovely* which is a signal-carrying word for sentiment detection) could be mistakenly removed through random deletion. A more advanced method is synonym insertion or replacement (Zhang et al., 2015; Wang and Yang, 2015), which uses Word2Vec (Mikolov et al., 2013) to replace words with their synonyms. Such a method respects the original sentence structure but fails to consider the context. It sometimes replaces words with synonyms that are awkward in the full context of the sentence. For example, replacing *lovely* with *fabulous* to get the sentence *"The baby is fabulous!"*. Recent work leans towards translation-based methods for augmentation (Fadaee et al., 2017; Silfverberg et al., 2017). In particular, Yu et al. (2018) proposed a back-translation method that first translates the text to French and then translates back to English, using the noisy output as the augmentation data. Although back-translation is intuitive and valid, its generation skews towards high frequency words (e.g., *cute, lovely* are both back-translated to *cute*), which not only causes repetition but also leads to lexical shrinkage in the augmented data. In a nutshell, existing techniques are still far from perfect, partially due to the strong interdependency of syntactic and semantic features in text data.

Original	So Cute! The baby is very lovely!
Naive Aug. *Delete + Swap*	So Cute! is The baby very!
Word2Vec Aug. *Insert + Replace*	So Cute *adorable*! The baby is very *fabulous*!
Back Translate Aug. *Eng. → Fr. → Eng.*	*Cute*! The baby is very *cute*!
Data Boost	***Look at this adorable baby!*** ***He is so cute!***

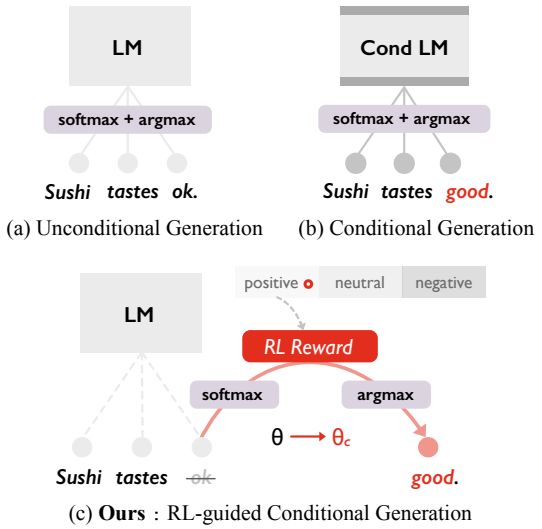
Table 1: A simple demo of existing text data augmentation methods on *positive* sentiment label.

In recent years, we have witnessed extensive progress in language models (LM). Large-scale LMs such as BERT (Devlin et al., 2019), XL-Net (Yang et al., 2019), and GPT-2 (Radford et al., 2019), are commonly trained on large amounts of text data (e.g., GPT-2 was trained on 8 million web pages that emphasized content diversity). One of the most interesting usages of these models is utilizing them as text generators (Raffel et al., 2019; Lewis et al., 2019; Dong et al., 2019). In this paper, we explore whether we can leverage the generation ability of the state-of-the-art LMs, to *generate* augmented samples for a given target class.

Augmentation samples should exhibit features of the target class. Off-the-shelf LMs cannot be directly used to augment data; since they are not trained for specific contexts, their generation is undirected and random. Conditional LMs can generate text directed by certain condition (e.g., target class), but they require training a LM from scratch with data covering all the conditions. Keskar et al. (2019), for instance, trained a 1.6 billion-parameter LM conditioned to a variety of control codes. The training is rather costly; however, collecting sufficient data for the training is also tedious, especially in low-resource tasks (Waseem, 2016).

We thus present **Data Boost**: a reinforcement learning guided text data augmentation framework built on off-the-shelf LM (GPT-2). Data Boost requires neither collecting extra data nor training a task-specific LM from scratch. We convert GPT-2 into a conditional generator, and for a given task, we guide the generator towards specific class labels during its decoding stage through reinforcement learning. The generated samples can then serve as augmentation data which are similar to the original data in terms of semantics and readability.

The advantages of Data Boost are three-fold: *First*, Data Boost is powerful. We achieve significant advances in three tasks on five different classifiers compared with six related works. *Second*, Data Boost generates sentence-level augmentation. Unlike prior methods that do word-level or phrase-level replacement (Kobayashi, 2018; Wei and Zou, 2019), our augmented data is of much greater variety in terms of vocabulary and sentence structure. Human evaluations also verify the high readability and label consistency of our augmentation. *Third*, Data Boost is easy to deploy. It does not require external datasets or training separate systems (like machine translation model in the back-translation method). Instead, we take the off-the-shelf GPT-2 language model and modify its decoding stage without changing its architecture.



(a) Unconditional Generation (b) Conditional Generation

(c) **Ours** : RL-guided Conditional Generation

Figure 1: General illustration of previous generation models and Data Boost. We add an additional RL stage between the softmax and argmax function, to update the LM hidden-states parameter θ towards target label (e.g. *positive*).

2 Data Boost

2.1 Conditional Generator

Given tokens $x_{<t} = \{x_0, x_1, ..., x_{t-1}\}$ and accumulated hidden states $h^{\theta}_{<t}$ [1] before time step t, a vanilla auto-regressive language model (LM) is trained to maximize probability of the next step token $\hat{x}_t$. Normally the model will pick the token that has the highest probability x_t as the t step decoding output:

$$x_t \sim \underset{\hat{x}_t}{\text{argmax}}\, p(\hat{x}_t|x_{<t}) = \text{LM}(x_{<t}, h^{\theta}_{<t}) \quad (1)$$

The generation of such step-by-step decoding is unconditional, since the model is trained on unannotated data (Figure 1 (a)). Conditional generation, however, normally needs to train a conditional LM. By modifying the LM architecture to allow for extra input (target label), the conditional LM can model the language and its corresponding label at the same time (Figure 1 (b)). Its generation is thus conditional on the label but the training of LM is always costly.

[1] Hidden states are basically key-values pairs in the attention blocks. We denote their values as parameter set θ.

Different from the above conditional generation method, we keep the architecture of the existing LM unchanged but postpone the argmax function to a later stage. In this way, the output of softmax is still differentiable (as it is a probability over the whole vocab rather than decoded tokens), which allows for gradient-based optimization. As shown in Figure 1 (c), we add a reinforcement learning (RL) stage within the gap between the softmax and argmax function. The RL reward (defined in Section 2.2.1) is where we inject the controlling signal of target label to guide the generation towards the target label. Specifically, in each decoding step, we update the hidden states parameter θ to the conditional θ_c in terms of the back-propagated reward after several iterations of RL optimization. The final decoded output shall be conditional on the target label (which is *positive* in Figure 1 (c)).

2.2 Reinforcement Learning Optimization

2.2.1 Reward

In the reinforcement learning framework, we define the state at step t as all the generated sequence before t (i.e., $s_t = x_{<t}$), and the action at step t as the t-th output token (i.e., $a_t = x_t$). The policy π_θ is interpreted as the probability we choose token x_t (action a_t) given the state $s_t = x_{<t}$, which is the softmax output of the hidden states (i.e., $\pi_\theta(a_t|s_t) = \text{softmax}(h^\theta_{<t})$, and similar for the conditional case).

We define the single-step reward of the conditional generated token x^c_t at step t as:

$$R(x^c_t) = \mathbb{E}_t\left[\frac{\pi_{\theta_c}(a_t|s_t)}{\pi_\theta(a_t|s_t)}G(x^c_t)\right] - \beta \text{KL}(\theta||\theta_c) \quad (2)$$

where $G(x^c_t)$ is the salience gain that measures how closely the generated token resembles the salient lexicon of the target label, and serves as a guide signal for the conditional generation. We also consider the Kullback–Leibler (KL) divergence between the conditional θ_c and unconditional distribution of θ as an auxiliary constraint (with weight β). Such a reward composition follows the classic PPO (Proximal Policy Optimization) (Schulman et al., 2017) form. Note that we are using an off-policy strategy to collect unconditional (s_t, a_t) pairs as trajectory to estimate the conditional reward $R(x^c_i)$. In this way, we are able to perform several iterations of updates on θ to maximize the reward without changing the sampling policy frequently, which avoids potential instability (Munos et al., 2016). As a result, we use the probability ratio between the conditional policy π_{θ_c} and the unconditional policy π_θ to re-calibrate the reward in the first term of Equation 2.

Salience Gain. For a given task that has K classes, we define the salience score of word x belonging to a certain class c as:

$$\mathcal{S}_{x,c} = \text{GM}\left(\frac{|x \in c|}{\sum_{k=1}^{K}|x \in c_k|}, \frac{|x \in c|}{\sum_{x_i \in |V|}|x_i \in c|}\right) \quad (3)$$

where $|x \in c|$ refers to the count of word x in samples with class label c, $|V|$ is the total vocabulary, and GM is geometric mean of the two terms. The two fractions try to guarantee that both $\mathbb{P}(c|x)$ and $\mathbb{P}(x|c)$ probabilities are high for a word marked as salient. We calculate the salience score for each word and pick the top-N highest words[2] as the salient lexicon for class label c (denoted as w_c). Compared with other methods such as training a discriminator (Dathathri et al., 2020) or deriving control codes (Keskar et al., 2019), we find our frequency-based method is relatively simple but efficient especially in data hungry cases, where the performance of a discriminator could be limited given very few training data.

For the t-th step token x^c_t conditional on the target class c, we calculate the salience gain as the logarithm summation of cosine similarity with each word in the salient lexicon w_c:

$$G(x^c_t) = \sum_{w_i \in w_c} \log(\text{softmax}(h^{\theta_c}_{<t}) \cdot \text{emb}(w_i)) \quad (4)$$

We use the embedded vector of w_i and the softmax output of t-th step hidden states $h^{\theta_c}_t$ to compute a dot product in the latent space. The salience gain measures how much the current step token resembles the salient lexicon of the target class.

KL Penalty. It is possible that the conditional policy π_{θ_c} drifts away too much from the unconditional policy π_θ resulting in an unreadable generation. Therefore, we incorporate a KL divergence penalty term measuring the distance between the

[2]N is a hyperparameter that is related to the size of the dataset. In our case, we set N=500 for sentiment analysis and irony classification tasks, and N=1000 for offense detection task.

two policies, in order to have better insurance that we are optimizing within a trust region. The KL divergence on the policies is computed as:

$$\mathrm{KL}(\theta||\theta_c) = \sum_{i\in[1,t]} \pi_\theta(a_i|s_i) \cdot \log \frac{\pi_\theta(a_i|s_i)}{\pi_{\theta_c}(a_i|s_i)} \quad (5)$$

We deduct KL divergence with weight β as a penalty term in the reward function (Equation 2). One can either choose a constant β or vary it dynamically.

2.2.2 Policy Gradient

Given the reward and the definitions described above, we update our policy at t-th step as:

$$\theta_c \leftarrow \theta + \eta \sum_{i=1}^{k} \frac{\nabla R(x_t^c/T)}{||\nabla R(x_t^c/T)||} \quad (6)$$

where η is the learning rate and θ_c is the parameter for the conditional hidden states. In general, we follow the classical SGD update rule, but make two main changes: (1) We use temperature parameter T to control the stochastic sampling during token decoding (Keskar et al., 2019). $T \to 0$ approximates a greedy decoding strategy that amplifies the peak in the vocab distribution while $T \to \infty$ makes the distribution more uniform. (2) We sum the normalized gradient of the reward for k steps. k can be treated as the strength of control over the conditional generation. Combining all above definitions, the policy gradient of Data Boost is summarized in Algorithm 1.

Algorithm 1: Data Boost Policy Gradient

Input: Target class label c, hidden-states param θ, target KL-divergence σ.
for $t = 0, 1, 2, \ldots$ **do**
  Generate $(a_t|s_t)$ by unconditional policy π_θ as trajectories;
  Estimate reward $R(x_t^c)$ using Eq. 2;
  Compute policy update using Eq. 6 by taking k steps of SGD (via Adam);
  if $KL(\theta||\theta_c) \geq 2\sigma$ **then**
    $\beta_{t+1} = 2\beta_t$;
  else if $KL(\theta||\theta_c) \leq \sigma/2$ **then**
    $\beta_{t+1} = \beta_t \,/\, 2$;
  end
  Return the conditional policy π_{θ_c};
end

We use a dynamic β to control the KL penalty within the reward function. The target divergence σ depends on the users' need: smaller σ means more resemblance to the unconditional generation while larger σ provides more space for RL guidance. After several iterations of RL optimization, the updated parameter set θ_c should be conditional on the target class label, whose feature lexicon contribute to the calculation of the reward R. We then use the conditional policy π_{θ_c} (which is based on the hidden states with θ_c) to decode this step token. The token should conform to the specified target class label c, since its corresponding hidden states have shifted towards c due to RL optimization.

3 Tasks & Datasets

We evaluated and compared Data Boost with several state-of-the-art text augmentation methods on the following three tasks:

Offense Detection[3] ICWSM 20' Data Challenge dataset ($N = 99,603$) for offensive language detection on tweets. The dataset consists of four classes: {*normal*, *spam*, *abusive* and *hateful*} with ratio {53.9%, 27.1%, 14.1%, 4.9%} respectively.

Sentiment Analysis[4] SemEval 2017 Task 4A dataset ($N = 20,631$) for sentiment analysis in tweets. There are three classes in the dataset: {*positive*, *neutral* and *negative*} with ratio {34.7%, 49.8%, 15.5%}.

Irony Classification[5] SemEval 2018 Task 3A dataset ($N = 3,817$) for irony detection in tweets. It has binary classes: {*ironic*, *non-ironic*}, with ratio {50.2%, 49.8%}.

Offense Detection and Irony Classification are popular NLU tasks that are low-resource. Sentiment Analysis, though seemingly well-resolved according to some literature (Baziotis et al., 2017; Cliche, 2017), is reported to have severe overfitting problems when given extremely limited training data (Elming et al., 2014; Severyn and Moschitti, 2015). We choose challenging datasets varying in total data size ($N \approx$ 80k, 17k, 3k) and the number of class (# of class $= 4, 3, 2$) for a realistic evaluation of our framework.

We removed all punctuation, stop words, hashtags and url links in the samples for all datasets. Samples whose length was above 30 tokens were filtered out (around 2% of the data on average) as

[3] https://sites.google.com/view/icwsm2020datachallenge
[4] http://alt.qcri.org/semeval2017/task4
[5] https://competitions.codalab.org/competitions/17468

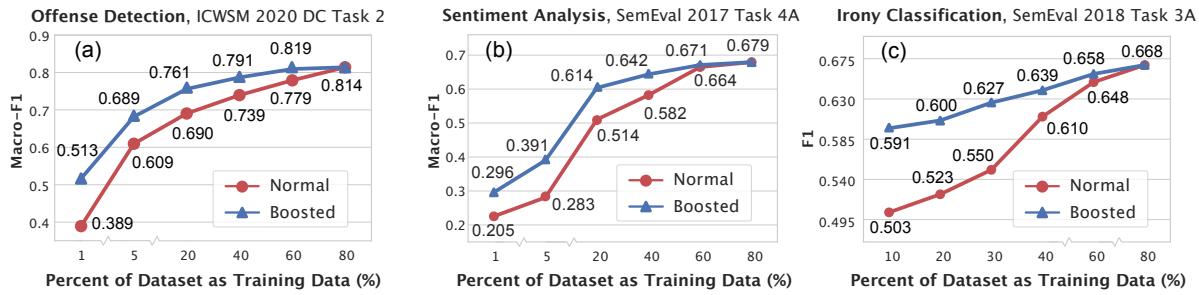


Figure 2: Performance with and without Data Boost on three classification tasks: (a) Offensive Language Detection on Tweets, (b) Sentiment Analysis in Twitter, (c) Irony Detection in English Tweets. The performance is reported by the Macro-F1 (F1 for binary task) of a BERT classifier averaged on five times repeated experiments.

30 was also used as the max sequence length for Data Boost generation. We further split the data into training and test set by the ratio {80%, 20%}, and maintained the original class distributions. We made sure the distributions remained the same in all of our experiments.

4 Experiments[6]

We conducted extensive experiments to answer the following three overarching questions about Data Boost:

4.1 Does Data Boost Improve Performance?

Several sets of data starvation tests are prepared, each using restricted fractions of the total data as training data. We keep test data the same (20% of the whole dataset) but gradually decrease the size of training data from 80% (as the fully-loaded case) to 1% (as the extremely low-resource case). We run both normal training and boosted training over the following training set fractions (%): {1%, 5%, 20%, 40%, 60%, 80%} of the total data for both Offense Detection and Sentiment Analysis. Since the dataset for Irony Classification is small ($N = 3,810$), we use the following fractions: {10%, 20%, 30%, 40%, 60%, 80%}. Note that for boosted training we add augmentation samples to training data until the training data size reaches 80% of the total size (same as fully-loaded size), to make sure that the size of the training set does not influence the results.

Figure 2 shows the performance of the BERT (Devlin et al., 2019) (bert-base-cased) classifier fine-tuned on the three tasks with and without Data Boost over all training set fractions. Data Boost has greater improvements on extremely low-resource cases: we achieve absolute F1 increases of 12.4% (Offense), 9.1% (Sentiment) and 8.8% (Irony) when using only 1% (10% for the Irony task) of the original data as training data. The results show that Data Boost can benefit a wide range of tasks with different characteristics. Also, since we used BERT as our classifier, which is already pre-trained on a large corpus, our results confirm that Data Boost can even improve the performance of large-scale LM based classifiers.

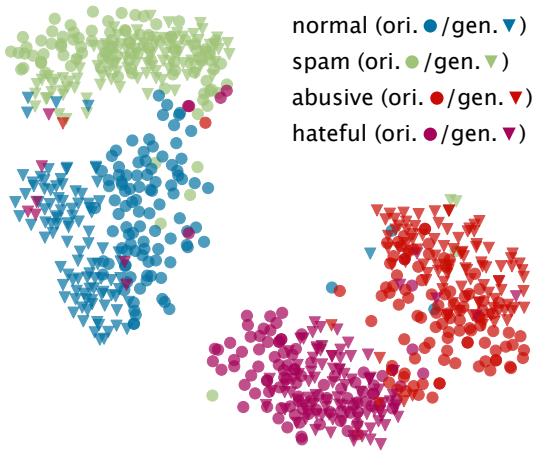


Figure 3: t-SNE visualization of the vectorized original and Data Boost augmented sentences in the offense detection task. The augmented sentences (triangles) mostly overlap with the original sentences (circles), suggesting that augmented sentences maintain the original class distribution.

4.2 Does Boosted Data Resemble the Original?

A common concern in text data augmentation is whether the augmented sentences preserve the qual-

[6]We run our generation and classification training on 2 RTX 2080 GPUs for all the experiments. The average time for Data Boost to generate a 30 token long sequence is under 1 second.

Classifier	Offense Detection			Sentiment Analysis			Irony Classification		
	20% (×2)	40% (×2)	80%	20% (×2)	40% (×2)	80%	20% (×2)	40% (×2)	80%
CNN (2014)	0.668	0.744	0.785	0.458	0.502	0.557	0.573	0.589	0.609
+ *Data Boost*	0.711	0.767	-	0.477	0.527	-	0.585	0.598	-
Bi-LSTM + Attn (2016)	0.696	0.744	0.788	0.439	0.515	0.564	0.468	0.554	0.598
+ *Data Boost*	**0.764**	0.778	-	0.513	0.542	-	0.550	0.579	-
Transformer (2017)	0.693	0.754	0.794	0.371	0.458	0.551	0.556	0.561	0.601
+ *Data Boost*	0.740	0.781	-	0.502	0.521	-	0.577	0.593	-
BERT (2019)	0.716	0.757	0.814	0.514	0.582	0.679	0.523	0.610	**0.668**
+ *Data Boost*	0.720	**0.784**	-	0.610	0.642	-	0.596	0.639	-
XLNet (2019)	0.680	0.718	**0.834**	0.624	0.643	**0.697**	0.632	0.639	0.664
+ *Data Boost*	0.693	0.755	-	**0.636**	**0.657**	-	**0.642**	**0.662**	-

Table 2: The classifier-agnostic experiments for five main-stream classifiers. We show the results before and after we apply Data Boost on two settings of training data: 20% original + 20% boosting data, and 40% original + 40% boosting data. We also list the performance of 80% as training data (full) as reference.

Offense Detection				
ratio	100% / 0	75% / 25%	50% / 50%	25% / 75%
F1	0.814	↓0.005	↓0.028	↓0.060
PPL	12.28	14.03	17.71	21.53
Sentiment Analysis				
ratio	100% / 0	75% / 25%	50% / 50%	25% / 75%
F1	0.679	↓0.008	↓0.037	↓0.065
PPL	22.52	25.76	29.41	36.01
Irony Classification				
ratio	100% / 0	75% / 25%	50% / 50%	25% / 75%
F1	0.668	↓0.010	↓0.029	↓0.068
PPL	33.47	36.85	40.71	45.53

Table 3: Evaluation of the generation quality in terms of F1 deterioration and perplexity (PPL) increase. We keep the training data size the same, but control the ratio of original/boosting. The first results column corresponds to no boosting.

ity of the original data. This is especially true for generation-based methods since we create new sentences rather than simply replace tokens to produce augmented data. We will illustrate the quality of our data generation with two approaches: (1) Visualizing the class distribution of the original and the augmented data (2) By using the boosting ratio experiments described in Section 4.1 to see whether data augmentation causes performance deterioration and perplexity increase.

For visualization, we randomly pick 400 (100 for each class) original and generated sentences in the Offense Detection task (since it has the largest number of classes) and vectorize with Sentence-BERT (Reimers and Gurevych, 2019). We apply t-SNE (Maaten and Hinton, 2008) to these vectors and plot their 2-D representations in Figure 3. From the figure, we can see that our RL-based algorithm manages to guide the generation towards the target labels, and for the most part, the distribution of generated sentences matches that of the original data.

Ratio-controlled experiments test the quality of boosted data by comparing training performance. If training on augmented dataset has comparable performance (F1) as training on purely original data, one may infer that the quality of the augmentation data resembles that of the original data. We also use perplexity (PPL) as an auxiliary metric to evaluate the augmentation quality. We trained three language models using kenLM (Heafield, 2011) on the original data of the three tasks. We use these models to calculate the perplexity of the ratio-controlled sets.

In Table 3 we show the F1 deterioration and perplexity increase (higher perplexity means poorer fit to the LM) for different augmentation ratios. Even when we use 25% original data fused with 75% generated samples, the F1 score only undergoes a slight decrease (0.06, absolute) compared to when using 100% original data. We found that the perplexity also did not substantially increase even with higher boosting ratios.

4.3 Is Data Boost Classifier-Agnostic?

We have shown Data Boost to be effective when used in conjunction with a BERT classifier, but can the performance be replicated with other clas-

Methods	Offense Detection			Sentiment Analysis			Irony Classification		
	10%	40%	PPL	10%	40%	PPL	10%	40%	PPL
Naive Aug. (Coulombe, 2018) (keyboard / OCR / spelling error)	0.670	0.661	209.17	0.566	0.609	318.99	0.567	0.532	195.81
Word Replace Aug. (Niu and Bansal, 2018) (synonyms + antonym from WordNet)	0.675	0.663	41.43	0.585	0.606	63.17	0.511	0.572	**57.52**
EDA (Wei and Zou, 2019) (randomly delete, swap, etc.)	0.637	0.629	37.37	0.560	0.608	**41.22**	0.530	0.515	76.07
Word2Vec Aug. (Wang and Yang, 2015) (insert, replace using Word2Vec)	0.673	0.720	376.43	0.557	0.619	561.31	0.548	0.585	384.61
Contextual Word Embs Aug. (Kobayashi, 2018) (insert, replace using Bi-RNN LM)	0.663	0.713	1729.62	0.610	0.627	1043.18	0.518	0.593	1146.40
Back-Translation Aug. (Yu et al., 2018) (Eng. $\rightarrow$ Fr. $\rightarrow$ Eng. as aug. text)	0.655	0.724	345.23	**0.617**	0.620	474.29	0.520	0.541	423.32
Ours: Data Boost (RL-guided conditional generation)	**0.695**	**0.784**	**35.18**	0.591	**0.642**	56.23	**0.591**	**0.639**	77.40

Table 4: Performance comparison with other text augmentation methods. 10%: 10% original data + 30% augmented data; 40%: 40% original data + 40% augmented data. We report the F1 score of the BERT classifier over five times repeat experiments. We also report the perplexity score (PPL) of the augmented data (10,000 randomly sampled) from different methods scored by kenLM language models trained on the training data of each task.

sifiers? In other words, is Data Boost a classifier-agnostic augmentation method? To answer this question, we ran experiments on four other mainstream classifiers, including the plain CNN classifier (Kim, 2014), the Bi-LSTM with attention mechanism (Zhou et al., 2016), the self-attention based Transformer network (Vaswani et al., 2017), and another LM-based classifier XLNet (Yang et al., 2019) for comparison. We trained all classifiers on three different training data settings: {20%, 40%, 80%} of the total data used as training data, the first two datasets are doubled in size using Data Boost augmentation. As shown in Table 2, Data Boost generally improves the performance of all the classifiers (from 1% to 13%, absolute), regardless of the classifier architecture. Moreover, we find Data Boost is not only effective for relatively simple classifiers (e.g., CNN), but also beneficial to complex LM-based classifiers (e.g., BERT and XLNet), which are already trained on a large corpus and generally used as very strong baselines for text classification tasks.

Table 5 shows sample generations by Data Boost.

5 Comparison with Related Work

Table 4 compares the performance of Data Boost with six prior text augmentation methods on all three tasks and using a BERT classifier. Naive methods (Coulombe, 2018; Xie et al., 2017) and translation-based methods (Fadaee et al., 2017; Sennrich et al., 2016) treated data noise either from artificial typos or translation errors as augmentation. Wei and Zou (2019) proposed EDA which is a combination of token-level augmentation (randomly delete, swap, etc.); they reported modest improvement (0.8% on average) on several benchmark datasets. Zhang et al. (2015) performed character-level augmentation. These methods were usually compromised by low readability and flawed syntactic structure. Other methods utilized external resources to improve augmentation quality. For example, Wang and Yang (2015) leveraged Word2Vec to extract synonyms. Kobayashi (2018) trained a Bi-RNN LM to propose replacements that are context-aware. Our tests find that these methods have higher perplexity than others. The reason could be that Word2Vec does not take context into account, while LM replacement highly depends on the quality of self-trained LM. Data Boost, however, is built on a state-of-the-art LM (GPT-2) and generates augmentations from scratch using RL rather than by replacement. Data Boost outperforms the other methods in the majority of the experiments (Table 4).

A few words about conditional generation: CTRL (Keskar et al., 2019) and BART (Lewis et al., 2019) are large-scale conditional LMs trained

Class	Generation Samples
ironic	freezing cold winter air can be a real treat but if your room temperature stays below freezing for a long time then the best way to cool down is death.
non-ironic	FoxNewscom reporter Michelle Fields is being sued by a Republican donor for disrespecting him and his family the Republican National Committee announced Wednesday.
positive	Congratulations to our friends at Bored Panda Pizza for the wonderful promotion that they have done We are very happy and proud to be able to share.
neutral	Results of the study revealed that the amount of protein ingested was similar in each group but not significantly different in total fat total carbohydrate or total protein.
negative	disappointed by the news media reports In the United States media have been covering reports on the killing of a woman by two men on a train.
normal	Im not a doctor or any other medical profession Im just trying to make this post useful to others who are looking through this topic.
spam	Black Friday sales on Xbox One begin today Nov at am ET Heres everything you can find in Black.
abusive	sick of all the crap If youve been following the news you know that the Trump administration and Democrats have been attacking President Trump executive.
hateful	idiot how does she know that you are fucking with her I dont want to see a stupid person like you get raped by any fucking person.

Table 5: Sample generation of Data Boost for all classes from three tasks. Salient words are underlined.

on self-collected data. PPLM (Dathathri et al., 2020) does conditional generation through perturbing the vocabulary distribution during token decoding. These methods have not been explored for text augmentation applications. More importantly, they do not use reinforcement learning to have fine-grained control over generation, which we found especially helpful when dealing with multiple labels within the same task.

6 Human Evaluation

6.1 Experimental Design

We conducted human evaluation on Amazon Mechanical Turk (MTurk) in May 2020. Participants (N = 178) were randomly assigned to evaluate one of the three tasks, respectively Irony Classification (n = 60), Sentiment Analysis (n = 58), and Offense Detection (n = 60). Participants were all from the United States and above 18 years old. The average age of participants was 36.92 years-old. More than half (57.3%) of participants were male, and 42.1% were female, one participant self-report gender as other. Each participant was paid 75 cents for their participation in this study.

6.2 Procedures

For each class, participants were asked to read three samples from each version (the original, unconditionally generated (vanilla GPT-2), and RL-conditional generated(Data Boost)). They were not informed of actual labels and versions of samples. After reading, participants were shown the actual label and version of those samples they just read. They were then asked to answer a series of questions about label agreement (e.g., *"How much do you agree with the assigned class label?"* on a 7-point scale (1-strongly disagree to 7-strongly agree)). Additionally, they were asked to rate the readability of samples on a 7-point scale (lower scores correspond to lower readability and vice versa). The readability measure included five items adapted from previous studies (Graefe et al., 2018), namely well-written, concise, comprehensive, coherent, and clear.

6.3 Results

6.3.1 Label Agreement

We conducted paired sample t-tests to examine how much participants agreed with the assigned labels. To conduct an ablation study, we included samples generated using vanilla GPT-2 and Data Boost. Compared to the vanilla GPT-2, Data Boost samples received higher label agreement scores in eight out of nine classes. Five of which were statistically significantly ($p < .05$) higher. No statistically significant differences were seen between the original

Task	Class	Mean (SD) Original	Data Boost	Vanilla	F	df	p-value
Irony Classification	*ironic*	4.65 (1.27)	5.04 (1.16)	5.04 (1.10)	2.21	177	0.11
	non-ironic	4.76 (1.31)	4.96 (1.17)	4.74 (1.36)	0.53	177	0.59
Sentiment Analysis	*positive*	4.92 (1.10)	4.90 (1.17)	4.51 (1.38)	2.03	171	0.13
	neutral	4.26 (1.39)	4.83 (1.23)	4.88 (1.16)	4.40	171	0.01**
	negative	4.41 (1.46)	4.72 (1.19)	4.94 (1.06)	2.61	171	0.08
Offense Detection	*hateful*	4.35 (1.51)	4.53 (1.43)	5.00 (1.27)	3.42	177	0.04*
	abusive	4.42 (1.38)	4.61 (1.21)	5.05 (1.19)	3.82	177	0.02*
	spam	4.43 (1.60)	4.86 (1.17)	4.63 (1.40)	1.39	177	0.25
	normal	4.83 (1.25)	5.15 (1.04)	4.92 (1.27)	1.11	177	0.33

Table 6: Human evaluation results on readability. p-value describes the significance of difference. (* corresponds to $p < 0.05$, ** to $p < 0.01$ and *** to $p < 0.001$.)

and boosted data, except for the *spam* and *normal* class in Offense Detection ($p = .02$ and $p = .03$). This result further confirms that Data Boost samples look very similar to the original samples and that Data Boost generates higher quality samples than the vanilla GPT-2.

6.3.2 Readability

We conducted several one-way analyses of variance (ANOVA) to test whether there were any statistically significant differences in the readability of the three models (Table 6). There were no significant differences for six of the classes. Curiously, for the *neutral* (Sentiment), *abusive* (Offense) and *hateful* (Offense) labels, both Data Boost and vanilla GPT-2 generated samples were rated as more readable than the original samples ($p < .05$). This could be explained by the fact that original samples are generally noisy tweets. These results indicate that the Data Boost generation has similar readability as the vanilla GPT-2 or original samples.

7 Limitations

In this section we discuss the limitations of Data Boost. The performance gain achieved by using Data Boost could be marginal on certain tasks, especially those whose classes cannot be modeled well by lexical features. For example, we experimented with Data Boost for metaphor detection using the LCC dataset (Mohler et al., 2016), sarcasm classification using the GHOSH dataset (Ghosh and Veale, 2017), and formality detection using the GYAFC formality style transfer dataset (Rao and Tetreault, 2018). We saw marginal improvements in the tasks, with an absolute increase in F1 scores of 1.3%, 0.9%, and 0.7% for the three tasks respectively (in the extreme data scarcity case, where we expect Data Boost to help the most; i.e., when boosting 1% of the original data to 80%). We found that it was difficult for our model to extract explicit lexical features for the *metaphor*, *sarcastic*, and *formal* classes. This could be because syntactic features play a role in these classes. It is challenging for Data Boost to compose meaningful augmentation in such cases, given that our guidance on the generation is token-by-token.

8 Conclusion

We have proposed a powerful and easy to deploy approach to augment text data through conditional generation. By leveraging an off-the-shelf language model (GPT-2), we successfully guide the generation towards a specified direction (i.e, target class), with the help of reinforcement learning. We find that Data Boost improves the performance of classification tasks, is classifier-agnostic, and that it surpasses several prior augmentation methods in three diverse classification tasks.

In the future, we plan to implement a more sophisticated guidance for the augmentation by adding syntactic and position features to the reward function, to enable augmentation of more diverse types of text data. The code will be made available upon request.

Acknowledgement

We sincerely thank the reviewers for their insightful comments and suggestions that helped improve the paper. This research was supported in part by the Dartmouth Burke Research Initiation Award and the Amazon Research Award.

References

Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. Datastories at semeval-2017 task 4: Deep lstm with attention for message-level and topic-based sentiment analysis. In *Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017)*, pages 747–754.

Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*.

Mathieu Cliche. 2017. Bb_twtr at semeval-2017 task 4: Twitter sentiment analysis with cnns and lstms. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 573–580.

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

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, pages 13042–13054.

Jakob Elming, Barbara Plank, and Dirk Hovy. 2014. Robust cross-domain sentiment analysis for low-resource languages. In *Proceedings of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 2–7.

Marzieh Fadaee, Arianna Bisazza, and Christof Monz. 2017. Data augmentation for low-resource neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 567–573.

Aniruddha Ghosh and Tony Veale. 2017. Magnets for sarcasm: Making sarcasm detection timely, contextual and very personal. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 482–491.

Andreas Graefe, Mario Haim, Bastian Haarmann, and Hans-Bernd Brosius. 2018. Readers' perception of computer-generated news: Credibility, expertise, and readability. *Journalism*, 19(5):595–610.

Kenneth Heafield. 2011. Kenlm: Faster and smaller language model queries. In *Proceedings of the sixth workshop on statistical machine translation*, pages 187–197. Association for Computational Linguistics.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. 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 452–457.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Michael Mohler, Mary Brunson, Bryan Rink, and Marc Tomlinson. 2016. Introducing the lcc metaphor datasets. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4221–4227.

Rémi Munos, Tom Stepleton, Anna Harutyunyan, and Marc Bellemare. 2016. Safe and efficient off-policy reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 1054–1062.

Tong Niu and Mohit Bansal. 2018. Adversarial oversensitivity and over-stability strategies for dialogue models. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 486–496.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 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. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Sudha Rao and Joel Tetreault. 2018. Dear sir or madam, may i introduce the gyafc dataset: Corpus, benchmarks and metrics for formality style transfer. 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 129–140.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. 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 3973–3983.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96.

Aliaksei Severyn and Alessandro Moschitti. 2015. Twitter sentiment analysis with deep convolutional neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 959–962.

Miikka Silfverberg, Adam Wiemerslage, Ling Liu, and Lingshuang Jack Mao. 2017. Data augmentation for morphological reinflection. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 90–99.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

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.

Zeerak Waseem. 2016. Are you a racist or am i seeing things? annotator influence on hate speech detection on twitter. In *Proceedings of the First Workshop on NLP and Computational Social Science*, pages 138–142, Austin, Texas. Association for Computational Linguistics.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification 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 6383–6389.

Ziang Xie, Sida I Wang, Jiwei Li, Daniel Lévy, Aiming Nie, Dan Jurafsky, and Andrew Y Ng. 2017. Data noising as smoothing in neural network language models. *arXiv preprint arXiv:1703.02573*.

Yiben Yang, Chaitanya Malaviya, Jared Fernandez, Swabha Swayamdipta, Ronan Le Bras, Ji-Ping Wang, Chandra Bhagavatula, Yejin Choi, and Doug Downey. 2020. G-daug: Generative data augmentation for commonsense reasoning. *arXiv*, pages arXiv–2004.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5754–5764.

Adams Wei Yu, David Dohan, Quoc Le, Thang Luong, Rui Zhao, and Kai Chen. 2018. Fast and accurate reading comprehension by combining self-attention and convolution. In *International Conference on Learning Representations*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 2: Short papers)*, pages 207–212.