Back-Training excels Self-Training at Unsupervised Domain Adaptation of Question Generation and Passage Retrieval

Devang Kulshreshtha[1,2], **Robert Belfer**[2], **Iulian Vlad Serban**[2], **Siva Reddy**[1]
[1]Mila/McGill University
[2]Korbit Technologies
{devang.kulshreshtha, siva.reddy}@mila.quebec, {iulian, robert}@korbit.ai

Abstract

In this paper, we propose a new domain adaptation method called *back-training*, a superior alternative to self-training. While self-training results in synthetic training data of the form quality inputs aligned with noisy outputs, back-training results in noisy inputs aligned with quality outputs. Our experimental results on unsupervised domain adaptation of question generation and passage retrieval models from *Natural Questions* domain to the machine learning domain show that back-training outperforms self-training by a large margin: 9.3 BLEU-1 points on generation, and 7.9 accuracy points on top-1 retrieval. We release *MLQuestions*, a domain-adaptation dataset for the machine learning domain containing 50K unaligned passages and 35K unaligned questions, and 3K aligned passage and question pairs. Our data and code is available at https://github.com/McGill-NLP/MLQuestions.

Algorithm	**Synthetic Training Data** Input	Output
Question Generation (QG)		
Self-Training	$p_u \sim P_\mathcal{T}(p)$	$\hat{q} \sim P_\mathcal{S}(q\|p_u)$
Back-Training	$\hat{p} \sim P_\mathcal{S}(p\|q_u)$	$q_u \sim P_\mathcal{T}(q)$
Passage Retrieval (IR)		
Self-Training	$q_u \sim P_\mathcal{T}(q)$	$\hat{p} \sim P_\mathcal{S}(p\|q_u)$
Back-Training	$\hat{q} \sim P_\mathcal{S}(q\|p_u)$	$p_u \sim P_\mathcal{T}(p)$

Table 1: *Self-Training* and *Back-Training* for unsupervised domain adaptation of question generation and passage retrieval. In self-training, inputs are sampled from the target domain data distribution $P_\mathcal{T}$ and their corresponding outputs are generated using a supervised model $P_\mathcal{S}$ trained on the source domain. In back-training, the inverse happens: outputs are sampled from $P_\mathcal{T}$ and their corresponding inputs are generated using $P_\mathcal{S}$. Notation: q and p denote questions and passages respectively, $._u$ denotes samples from the target domain and $\hat{.}$ denotes the samples generated by a supervised model trained on the source domain.

1 Introduction

In domains such as education and medicine, collecting labeled data for tasks like question answering and generation require domain experts, thereby making it expensive to build supervised models. Transfer learning can circumvent this limitation by exploiting models trained on other domains where labeled data is readily available (Bengio, 2012; Ruder et al., 2019). However, using these pretrained models directly without adapting to the target domain often leads to poor generalization. To address this issue, these models are further trained on cheap data such as synthetically generated labeled data for the target domain by exploiting unsupervised data from the target domain (Ramponi and Plank, 2020). One such popular data augmentation method for unsupervised domain adaptation is *self-training* (Yarowsky, 1995).

In self-training, given a pretrained model that can perform the task of interest in a source domain and unlabeled data from the target domain, the pretrained model is used to predict the labels for the target domain data. The pretrained model is further trained on the synthetic data to adapt to the new domain (this step is also known as fine-tuning for domain adaptation). To improve the quality of the synthetic data, it is also common to use consistency checks such as filtering data with low model confidence.

In this paper we propose a new method called *back-training* a superior alternative to self-training for domain adaptation (the name is inspired from *back-translation* for machine translation). While self-training results in synthetic data of the form quality inputs aligned with noisy outputs, back-training results in noisy inputs aligned with quality outputs.

We focus on domain adaptation for question generation and passage retrieval. Our domain of interest is *machine learning* as it is a rapidly evolving area of research, and question generation and passage retrieval tasks could empower student learning on MOOCs. For example, from a passage about linear and logistic regression, an education bot could generate a question such as *what is the difference between linear and logistic regression?* to teach a student about these concepts. Moreover, passage retrieval models could help students to find relevant passages for a given question. In this domain, unsupervised data such as text passages and questions are easy to obtain than their alignments with each other.

We adapt question generation and passage retrieval models trained on generic domains such as Wikipedia to our target domain.

We first show that without any domain adaptation, these models perform poor on the target domain. While self-training improves the domain performance, we find that back-training outperforms self-training by a large margin.

Table 1 demonstrates the differences between self-training and back-training for question generation and passage retrieval. Consider the question generation task. For self-training, we first train a supervised model $P_S(q|p)$ on the source domain that can generate a question q given a passage p. We use this model to generate a question $\hat{q}$ for an unsupervised passages p_u sampled from the target domain distribution $P_T(p)$. Note that $\hat{q}$ is generated conditioned on the target domain passage using $P_S(q|p_u)$. We use the pairs (p_u, $\hat{q}$) as the synthetic training data to adapt $P_S(q|p)$ to the target domain. In back-training, we assume access to unsupervised questions and passages from the target domain. We first train a passage retrieval model $P_S(p|q)$ on the source domain. We then sample a question q_u from the target domain distribution $P_T(q)$. We condition the retriever on this question i.e., $P(p|q_u)$, and retrieve a passage $\hat{p}$ from the target domain and treat it as a noisy alignment. We use the pairs ($\hat{p}$, q_u) as the synthetic training data to adapt $P_S(q|p)$. Table 1 also describes the details of domain adaptation for the passage retriever.

Our contributions and findings are as follows: 1) We release *MLQuestions* a domain adaptation dataset for the machine learning domain containing 50K unaligned passages and 35K unaligned questions, and 3K aligned passage and question pairs. 2) We show that question generation and passage retrieval models trained on NaturalQuestions (Kwiatkowski et al., 2019) generalize poorly to machine learning domain, with at least 20 points drop in performance on both BLEU1 for generation and R@1 for retrieval. 3) Although self-training improves the domain performance marginally, our back-training method outperforms self-training with 7-10 points. 4) We further improve the performance back-training with consistency filters and iterative refinement.

2 Background

In this section, we describe the source and target domain datasets, the models for question generation and passage retrieval, and the evaluation metrics.

2.1 Source Domain: NaturalQuestions

We use NaturalQuestions (Kwiatkowski et al., 2019) dataset as our source domain. NaturalQuestions is an open-domain question answering dataset containing questions mined from Google search engine queries paired with answers from Wikipedia. We use the long form of the answer which corresponds to passages (paragraphs) of Wikipedia articles. It is the largest dataset available for open-domain question answering, comprising of 300K training examples, each example comprising of a question paired with a Wikipedia passage. We label 200 random questions of NaturalQuestions and annotate them into 5 different classes based on the nature of the question. Table 2 shows these classes and their distribution. As seen, 86% of the questions are descriptive questions that start with *what, who, when* and *where*.

2.2 Target Domain: Machine Learning

Our target domain of interest is machine learning. There is no large supervised dataset for question answering for this domain, and it is expensive to create one since it requires domain experts. However, it is relatively cheap to collect large number of machine learning articles and questions. We collect machine learning concepts and passages from the Wikipedia Machine Learning category

Taxonomy	Description *(Frequent Wh-words)*	Examples (from MLQuestions)	Distribution (%) NaturalQuestions	MLQuestions
DESCRIPTION	Asking definition or examples about a concept *(What, Who, When, Where)*	What is supervised learning with example?	86%	40%
METHOD	Computational or procedural questions - *(How)*	How do you compute vectors in Word2Vec?	1%	16%
EXPLANATION	Causal, justification or goal-oriented questions - *(Why)*	Why does ReLU activation work so surprisingly well?	3%	19%
COMPARISON	Ask to compare more than one concept with each other	What is the difference between LDA and PCA?	5%	11%
PREFERENCE	Yes/No or select from valid set of options - *(Is, Are)*	Is language acquisition innate or learned?	5%	19%

Table 2: Classification of 200 random questions from NaturalQuestions and MLQuestions.

page[1] and recursively traversing its subcategories. We end up with 17K distinct concepts such as *Autoencoder*, *word2vec* etc. and 50K passages.

For question mining, we piggy-back on Google Suggest's *People also ask* feature to collect 104K questions by using machine learning concept terms as seed queries combined with question terms such as *what*, *why* and *how*. However many questions could belong to generic domain due to ambiguous terms such as *eager learning*. We employ three domain experts to annotate 1000 questions to classify if a question is in-domain or out-of-domain. Using this data, we train a classifier (Liu et al., 2019) to filter questions that have in-domain probability less than 0.8. This resulted in 47K in-domain questions, and has 92% accuracy upon analysing 100 questions. Of these, we use 35K questions as unsupervised data.

The rest of the questions are used to create supervised data for model evaluation. We use Google search engine to find answer passages to these questions, resulting in 11K passages. Among these, we select 3K question and passage pairs as the evaluation set for question generation (50% validation and 50% test). For passage retrieval, we use the full 11K passages as candidate passages for the 3K questions. We call our dataset *MLQuestions*.

Table 2 compares MLQuestions with NaturalQuestions. We note that MLQuestions has higher diversity of question classes than NaturalQuestions, making the transfer setting challenging.

[1] https://en.wikipedia.org/wiki/Category:Machine_learning

2.3 Question Generation Model

We use BART (Lewis et al., 2020) to train a supervised question generation model on NaturalQuestions. BART is a Transformer encoder-decoder model pretrained to reconstruct original text inputs from noisy text inputs. Essentially, for the question generation task, BART is trained to learn a conditional language model $P_S(q|p)$ that generates a question q given a passage p from the source domain. For more experimental details, see Appendix A.1.

2.4 Passage Retrieval Model

We use the pretrained Dense Passage Retriever (DPR; Karpukhin et al. 2020) on NaturalQuestions. DPR encodes a question q and passage p separately using a BERT bi-encoder and is trained to maximize the dot product (similarity) between the encodings $E_P(p)$ and $E_Q(q)$, while minimizing similarity with other closely related but negative passages. Essentially, DPR is a conditional classifier $P_S(p|q)$ that retrieves a relevant passage p given a question q from the source domain.

2.5 Evaluation Metrics

We evaluate question generation using standard language generation metrics: BLEU1-4 (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005) and ROUGE$_L$ (Lin, 2004). For passage retrieval, we report Top-K retrieval accuracy for $K = 1, 20, 40, 100$ following Karpukhin et al. (2020) by measuring the fraction of cases where the correct passage lies in the top k retrieved passages. We consider all 11K passages in MLQuestions for retrieval during test time.

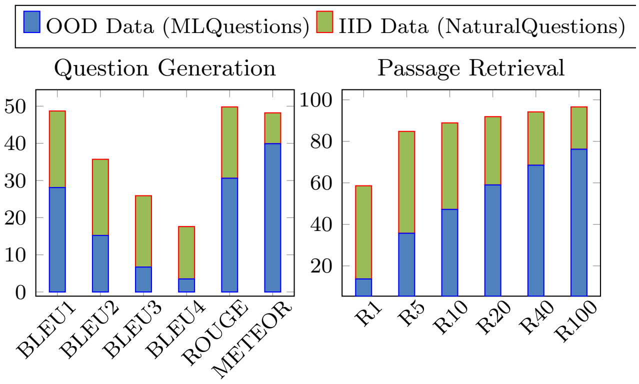

Figure 1: IID/OOD generalization gaps for Question Generation and Passage Retrieval. The green+blue bar measures performance on NaturalQuestions (source) domain, while green bar measures performance drop due to domain shift to MLQuestions (target) domain.

3 Transfer from Source to Target Domain without Adaptation

We investigate how well models trained on NaturalQuestions transfer directly to MLQuestions without any domain adaptation. For comparison, we also present the results on NaturalQuestions. To be fair, we sample equal number of samples from the development set of NaturalQuestions as in the test set of MLQuestions for question generation and passage retrieval. Figure 1 shows the results. We observe high performance drops across all generation metrics (14-20%) from NaturalQuestions (IID data) to MLQuestions (OOD Data). Human evaluation on question generation (see Table 7) also reveals that the generated questions are either generic, or fail to understand domain-specific terminology. OOD performance in the passage retrieval task is even worse (25-40% drop), revealing a huge distribution shift between the source and target domain.

4 Unsupervised Domain Adaptation

In this section, we describe self-training and back-training methods to generate synthetic training data for unsupervised domain adaptation (UDA). We also describe consistency filters which can further improve the quality of the synthetic data.

4.1 Problem Setup

The source domain consists of labeled data containing questions paired with passages $\mathcal{D}_\mathcal{S} \equiv \{(q_s^i, p_s^i)\}_{i=1}^m$. The target domain consists of unlabeled passages $\mathcal{P}_\mathcal{U} \equiv \{p_u^i\}_{i=1}^{m_p}$, and unlabeled

Notation	Definition
S, T	Source, Target Domain
P_S, P_T	Source, Target data distribution
$\mathcal{D}_\mathcal{S} \equiv \{(q_s^i, p_s^i)\}_{i=1}^m$	Source labeled corpus
$\mathcal{P}_\mathcal{U} \equiv \{p_u^i\}_{i=1}^{m_p}$	Target unlabeled passages
$\mathcal{Q}_\mathcal{U} \equiv \{q_u^i\}_{i=1}^{m_q}$	Target unlabeled questions
$\boldsymbol{\theta} \equiv \{\boldsymbol{\theta}_G, \boldsymbol{\theta}_R\}$	QG, IR Models
S_G, S_R	Psuedo-target data for QG, IR

Table 3: Notations used throughout the paper.

questions $\mathcal{Q}_\mathcal{U} \equiv \{q_u^i\}_{i=1}^{m_q}$. Note that $\mathcal{P}_\mathcal{U}$ and $\mathcal{Q}_\mathcal{U}$ are *not necessarily* aligned with each other. Given this setup, our goal is to learn question generation (QG) and passage retrieval (IR) models with parameters $\boldsymbol{\theta} \equiv \{\boldsymbol{\theta}_G, \boldsymbol{\theta}_R\}$ that can achieve high generation and retrieval performance on target domain T. Table 3 describes the notations used across the paper.

4.2 Self-Training for UDA

Self-training involves training a model on its own predictions. We present the proposed self-training for UDA in Algorithm 1. First the baseline models $\boldsymbol{\theta_G}$ and $\boldsymbol{\theta_R}$ are trained on the source passage-question corpus $\mathcal{D}_\mathcal{S}$. Then, at each iteration, the above models generate pseudo-labeled data from unlabeled passages $\mathcal{P}_\mathcal{U}$ for question generation and questions $\mathcal{Q}_\mathcal{U}$ for passage retrieval. For QG, $\boldsymbol{\theta_G}$ generates a question $\hat{q}$ for each $p_u \in \mathcal{P}_\mathcal{U}$ and adds $(p_u, \hat{q})$ to the synthetic training data S_G. For IR, $\boldsymbol{\theta_R}$ retrieves a passage $\hat{p}$ from $\mathcal{P}_\mathcal{U}$ for each $q_u \in \mathcal{Q}_\mathcal{U}$ and adds $(q_u, \hat{p})$ to S_R. The models $\boldsymbol{\theta_G}$ and $\boldsymbol{\theta_R}$ are fine-tuned on S_G and S_R respectively, and the same process is repeated for a desired number of iterations. Note that in self-training inputs are sampled from the target domain and the outputs are predicted (noisy).

4.3 Back-Training for UDA

The main idea of back-training is to work backwards: start with true output samples from the target domain, and predict corresponding inputs which aligns the most with the output. While self-training assumes inputs are sampled from the target domain distribution, back-training assumes outputs are sampled from the target domain distribution. When two tasks are of dual nature (i.e., input of one task becomes the output of another task), back-training can be used to generate synthetic training data of one task using the other, but on a condition that outputs can be sampled from

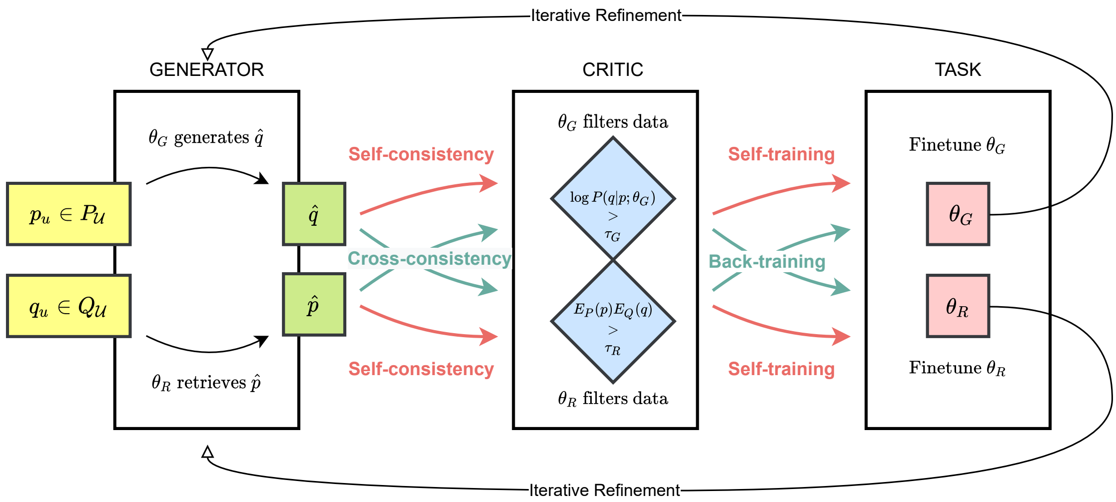

Figure 2: **Abstract representation of bootstrapping algorithms with consistency filtering:**. (1) The generator produces pseudo-data from unlabeled target data (2) The critic filters data produced by generator via **Self-consistency (SC)** or **Cross-consistency (CC)** filtering (3) Filtered data is used for training models via **Self-training (ST)** or **Back-training (BT)**. This combination gives us four model variants - **ST-SC, ST-CC, BT-SC, BT-CC**.

the target domain distribution. QG and IR tasks meet both criteria. For QG, we have unlabeled questions in the target domain and its dual friend IR can retrieve their corresponding input passages from the target domain, and for IR we have passages in the target domain and QG can generate their input questions. Formally, for QG, the IR model $\boldsymbol{\theta_R}$ retrieves passage $\hat{p}$ from $\mathcal{P}_\mathcal{U}$ for each $q_u \in \mathcal{Q}_\mathcal{U}$ and adds $(\hat{p}, q_u)$ to S_G. For IR, the QG model $\boldsymbol{\theta_G}$ generates a question $\hat{q}$ for each $p_u \in \mathcal{P}_\mathcal{U}$ and adds $(\hat{q}, p_u)$ to S_R.

Similarities with *back-translation* Back-translation is an effective method to improve machine translation systems using synthetic parallel corpora containing human-produced target language sentences paired with artificial source language translations (Sennrich et al., 2016; Edunov et al., 2018). Back-training is inspired by this idea, however it is not limited to machine translation.

4.4 Consistency filters for Self-Training and Back-Training

The above algorithms utilize the *full* unlabeled data along with their predictions even if the predictions are of low confidence. To alleviate this problem, in self-training, it is common to filter low-confidence predictions (Zhu, 2005). We generalize this notion as *consistency filtering*: For the

Algorithm 1 Vanilla Self-Training Back-Training for unsupervised domain adaptation. Vanilla algorithms can be improved further using consistency filters

Input: Source Data $\mathcal{D}_\mathcal{S} \equiv \{(q_s^i, p_s^i)\}_{i=1}^m$, Target unlabeled data $\mathcal{P}_\mathcal{U} \equiv \{p_u^i\}_{i=1}^{m_P}$, $\mathcal{Q}_\mathcal{U} \equiv \{q_u^i\}_{i=1}^{m_q}$
Output: Target domain QG model $\boldsymbol{\theta_G}$, IR model $\boldsymbol{\theta_R}$

```
Init: θ_G, θ_R ← Train on D_S
repeat
    S_G ← [ ], S_R ← [ ]    ▷ Pseudo-data for θ_G and θ_R
    for q ∈ Q_U do
        p̂ ← Retrieve p from P_U closest to q using θ_R
        add (p̂, q) to S_R  S_G
    end for
    for p ∈ P_U do
        q̂ ← Generate q from p using θ_G
        add (p, q̂) to S_G  S_R
    end for
    θ_G ← Finetune on S_G, θ_R ← Finetune on S_R
until dev performance decreases
```

tasks QG and IR, a *generator* $G \in \{\boldsymbol{\theta}_G, \boldsymbol{\theta}_R\}$ produces synthetic training data for a task whereas the *critic* $C \in \{\boldsymbol{\theta}_G, \boldsymbol{\theta}_R\}$ filters low confidence predictions. We define two types of consistency filtering: 1) **Self-Consistency (SC)** where the generator and critic are the *same*; and 2) **Cross-Consistency (CC)** where the generator and critic are *different*. These consistencies can be extended further to create a round trip (cycle) consistency (Alberti et al., 2019). Self-training and back-training can be combined with self-consistency or cross-consistency or both. Figure 2 shows these combi-

Model	Question Generation						Passage Retrieval			
	B1	B2	B3	B4	M	R	R@1	R@20	R@40	R@100
No-adaptation	28.14	15.29	6.79	3.50	39.94	30.69	13.77	59.02	68.51	76.27
Self-Training	29.24	16.12	7.15	3.73	41.56	31.96	14.73	62.59	71.85	81.54
Back-Training	**37.44**	**25.70**	**14.58**	**8.90**	**42.17**	**42.28**	**21.68**	**75.57**	**84.35**	**90.31**

Table 4: Results of unsupervised domain adaptation on MLQuestions. *No-adaptation* denotes the model trained on NaturalQuestions and tested directly on MLQuestions without any domain adaptation.

nations.

5 Domain Adaptation Evaluation

As described in Section 2, our source domain is NaturalQuestions and the target domain is MLQuestions. We evaluate if domain adaptation helps to improve the performance compared to no adaptation. We investigate the effectiveness of self-training and back-training and their qualitative differences. We also investigate if consistency filters and iterative refinement result in further improvements.

5.1 No-adaptation vs. self-training vs back-training

In Table 4, we compare the performance of vanilla self-training and back-training algorithms (i.e., without consistency filtering and no iterative refinement) with the no-adaptation baseline model (i.e. the model trained on source domain and directly tested on the target domain). We note that both self-training and back-training are superior to no-adaptation baseline. Self-Training achieves an absolute gain of around 1 BLEU-1 point for QG and 1 point in R@1 for IR. Whereas back-training vastly outperforms self-training, with improvements of 9.3 BLEU-1 points on QG and 8 points in R@1 on IR compared to the no-adaptation baseline.

5.2 Qualitative analysis of self-training and back-training

Figure 3 shows the perplexity of the QG models on synthetic training data and the test data as the training (domain adaptation) proceeds. The plot reveals three interesting observations: (1) for back-training, the train and test perplexity (and hence likelihood) are correlated and hence the data generated by back-training matches the target distribution more closely than self-training; (2) self-training achieves lower training error but higher test error compared to back-training, indicating overfitting; (3) extrapolating back-training curve suggests that scaling additional unlabeled data will likely improve the model.

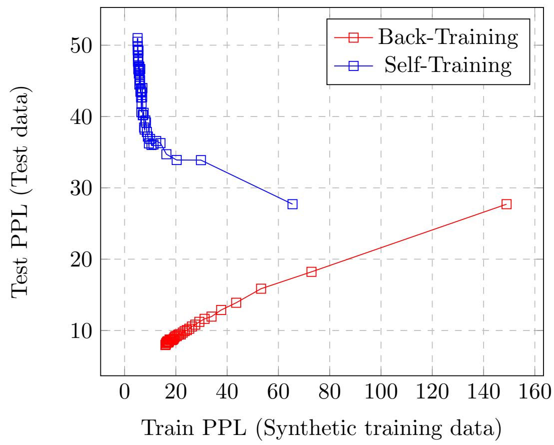

Figure 3: Evolution of QG model perplexity for Self-Training vs Back-Training as training proceeds. Trajectories run from right to left as perplexity decreases with training. Rightmost points are plotted after first mini-batch training, and perplexity is computed after each mini-batch training.

Figures 4 and 5 plot the distribution for self-training (computing scores of model's own predictions) and back-training (computing scores of different model's predictions) for QG and IR tasks. The figures reveal that self-training has *high mean* and *low variance*, indicating less diverse training data. On the other hand, back-training distribution has *low mean* and *high variance* indicating diverse training data.

5.3 Are consistency filters useful?

We use consistency filters to filter the synthetic training data. All model combinations are shown in Table 5. Our experiments are still in progress at the time of paper submission, but the preliminary results indicate consistency filters are promising.

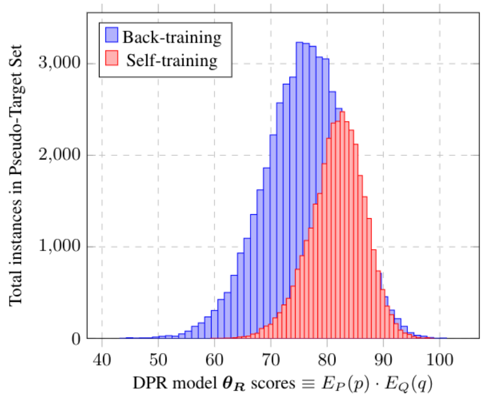

Figure 4: DPR score distribution for pseudo-data computed using $\boldsymbol{\theta_R}$: Self-training vs Back-training.

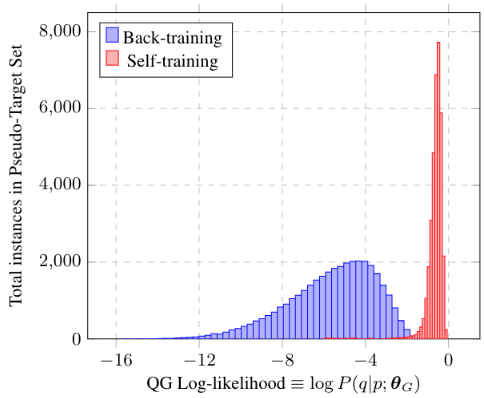

Figure 5: Log-likelihood distribution for pseudo-data computed using $\boldsymbol{\theta_G}$: Self-training vs Back-training.

5.4 Is iterative refinement useful?

Further performance improvement of up to 2.17 BLEU-1 points and 1.64 R@40 points can be observed in back-training (Table 6) via the iterative procedure described in Algorithm 1. On the other hand, self-training does not show any improvements for QG and marginal improvements for IR.

5.5 Human Evaluation Results

We also report human evaluation of QG by sampling 50 generated questions from the test set and asking three domain experts to rate a question as good or bad based on four attributes: *Naturalness*, i.e., fluency and grammatical correctness; *Coverage*, i.e., whether question covers the whole passage or only part of the passage; *Factual Correctness* in ML domain; *Answerability*, i.e., if the question can be answered using the passage. We report the average of ratings in in Table 7. We observe that the back-training model is superior on all four criteria. However, all models perform similarly on *naturalness*.

Task/ Consistency	BLEU-1 (QG)		R@40 (IR)	
	ST	BT	ST	BT
No consistency	29.24	37.44	71.85	84.35
Self consistency	-	38.73	-	84.85
Cross consistency	-	-	-	-

Table 5: Consistency performance on MLQuestions dataset for Self-Training (ST) and Back-Training (BT). - denotes experiment results are pending.

Task/ Iteration	BLEU-1 (QG)		R@40 (IR)	
	ST	BT	ST	BT
$T = 1$	29.24	37.44	71.75	84.35
$T = 2$	29.14	40.31	72.03	85.89
$T = 3$	29.08	40.25	71.98	85.97
Max Gain	0.0	2.17	0.28	1.64

Table 6: Evolution of model performance with increasing iterations: Blue numbers denote increases in performance, while Red denotes decreases in performance. Only Back-training benefits from iterative refinement.

5.6 Analysis of Question Types

We analyze how well our QG model can generate different kinds of questions according to the taxonomy described in Table 2. In Figure 6 we plot the confusion matrix between the actual question class and generated question class for our best model (BT-SC). To do this, 100 actual questions and corresponding generated questions are sampled from the MLQuestions test set and annotated by a domain expert. We find that the model generates few *Explanation* questions and even fewer *Preference* questions while over-generating *Description* questions. *Comparison* and *Method* questions show good F1-score overall, hence these classes benefit the most from domain adaptation.

6 Related Work

Question Generation Most literature on QG have focused on building supervised models using neural encoder-decoder models (Du et al., 2017; Mishra et al., 2020; Zhao et al., 2018; Chan and Fan, 2019; Klein and Nabi, 2019). Tang et al. (2018) propose to improve QG through QA-specific signal as the loss function, and improving QA model by generating synthetic data using QG model. Duan et al. (2017) generate QA pairs from YahooAnswers, and improve QA by adding a question-consistency loss in addition to QA loss.

Model	N	C	FC	A
No-adaptation	0.68	0.32	0.58	0.68
Self-Training	0.68	0.36	0.58	0.72
Back-Training	**0.70**	**0.44**	**0.62**	**0.88**

Table 7: Human evaluations scores between 0-1 on 50 model generated questions for four criteria: Naturalness (N), Coverage (C), Factual Correctness (FC), and Answerability (A).

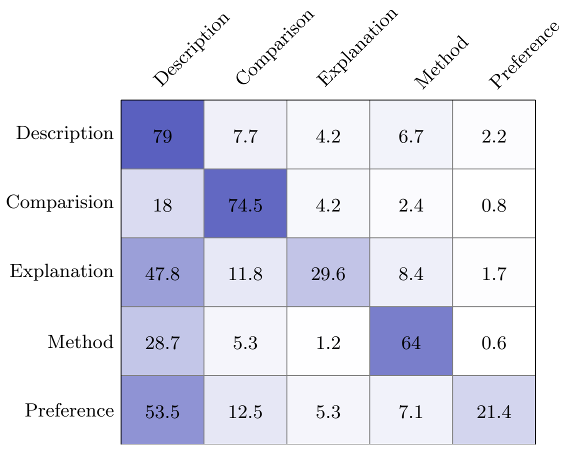

Figure 6: Confusion matrix of actual (row) vs model generated question (column) classes for 100 questions sampled from MLQuestions test set. Values are in % where each row sums to 100%.

To our knowledge, ours is the first work towards unsupervised domain adaptation for question generation.

Passage Retrieval has previously been performed using classical approaches like Lucene-BM25 (Robertson and Zaragoza, 2009) which represent question and passage as *sparse* vectors, and match keywords efficiently using TF-IDF. Recently, Karpukhin et al. (2020) show that fine-tuning dense representations of questions and passages from BERT can outperform classical methods by a strong margin. We adopt the same model for domain adaptation on MLQuestions dataset. Concurrent to our work, Reddy et al. (2020) also perform domain adaptation for passage retrieval. Our focus has been on systematically approaching the unsupervised domain adaptation for both QG and IR.

Bootstrapping methods such as self-training have been applied for numerous domains such as Computer Vision (Saito et al., 2020; Yang et al., 2021), ASR (Kahn et al., 2020; Pino et al., 2020). In NLP, self-training has been applied for question answering (Chung et al., 2018), machine translation (Ueffing, 2006), and sentiment analysis (He and Zhou, 2011). Sachan and Xing (2018) apply self-training to generate synthetic data for QA and QG in the same domain, and filter data using QA model confidence on answer generated by the question.

7 Conclusion and Future Work

We introduce back-training as an unsupervised domain adaptation method focusing on Question Generation and Passage Retrieval. Our algorithm generates synthetic data pairing high-quality outputs with noisy inputs in contrast to self-training producing noisy outputs aligned with quality inputs. We find that back-training outperforms self-training in target domain by a large margin on our newly released dataset *MLQuestions*. We further improve these models with consistency constraints and iterative learning.

One area of future research will be exploring back-training on interesting tasks. Visual Question Generation (Mostafazadeh et al., 2016) and Image Retrieval (Datta et al., 2008) are analogous to Question Generation and Passage Retrieval where back-training can be applied. We would also like to establish theoretical justifications for superior performance of back-training in context with generalization errors and distribution overlap between target domain data and synthetic data generated by back-training.

Acknowledgments

We thank the members of SR's research group for their constant feedback during the course of work. We thank Ekaterina Kochmar, Ariella Smofsky and Shayan from Korbit ML team for their helpful comments. SR is supported by the Facebook CIFAR AI Chair and DK is supported by the MITACS fellowship. We would like to thank SerpAPI for providing search credits for data collection in this work.

References

Chris Alberti, Daniel Andor, Emily Pitler, Jacob Devlin, and Michael Collins. 2019. Synthetic QA corpora generation with roundtrip consistency. In *Proceedings of the 57th Annual Meet-*

ing of the Association for Computational Linguistics, pages 6168–6173, Florence, Italy. Association for Computational Linguistics.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *ACL workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Yoshua Bengio. 2012. Deep learning of representations for unsupervised and transfer learning. In *Proceedings of ICML workshop on unsupervised and transfer learning*, pages 17–36. JMLR Workshop and Conference Proceedings.

Ying-Hong Chan and Yao-Chung Fan. 2019. Bert for question generation. In *12th International Conference on Natural Language Generation (INLG)*, pages 173–177.

Yu-An Chung, Hung-Yi Lee, and James Glass. 2018. Supervised and unsupervised transfer learning for question answering. In *North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 1585–1594.

Ritendra Datta, Dhiraj Joshi, Jia Li, and James Z Wang. 2008. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys (Csur)*, 40(2):1–60.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *55th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1342–1352.

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *Empirical Methods in Natural Language Processing(EMNLP)*, pages 866–874.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 489–500.

Angela Fan, David Grangier, and Michael Auli. 2018. Controllable abstractive summarization. *Association for Computational Linguistics (ACL)*, page 45.

Yulan He and Deyu Zhou. 2011. Self-training from labeled features for sentiment analysis. *Information Processing & Management*, 47(4):606–616.

Jacob Kahn, Ann Lee, and Awni Hannun. 2020. Self-training for end-to-end speech recognition. In *ICASSP IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7084–7088. IEEE.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Tassilo Klein and Moin Nabi. 2019. Learning to answer by learning to ask: Getting the best of gpt-2 and bert worlds. *arXiv preprint arXiv:1911.02365*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics (TACL)*, 7:453–466.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

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*.

Shlok Kumar Mishra, Pranav Goel, Abhishek Sharma, Abhyuday Jagannatha, David Jacobs, and Hal Daume. 2020. Towards automatic generation of questions from long answers. *arXiv preprint arXiv:2004.05109*.

Nasrin Mostafazadeh, Ishan Misra, Jacob Devlin, Margaret Mitchell, Xiaodong He, and Lucy Vanderwende. 2016. Generating natural questions about an image. In *Association for Computational Linguistics (ACL)*, pages 1802–1813.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Association for Computational Linguistics (ACL)*, pages 311–318.

Juan Pino, Qiantong Xu, Xutai Ma, Mohammad Javad Dousti, and Yun Tang. 2020. Self-training for end-to-end speech translation. *Proc. Interspeech 2020*, pages 1476–1480.

Alan Ramponi and Barbara Plank. 2020. Neural unsupervised domain adaptation in NLP—A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6838–6855, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Revanth Gangi Reddy, Bhavani Iyer, Md Arafat Sultan, Rong Zhang, Avi Sil, Vittorio Castelli, Radu Florian, and Salim Roukos. 2020. End-to-end qa on covid-19: Domain adaptation with synthetic training. *arXiv preprint arXiv:2012.01414*.

Stephen Robertson and Hugo Zaragoza. 2009. *The probabilistic relevance framework: BM25 and beyond*. Now Publishers Inc.

Sebastian Ruder, Matthew E Peters, Swabha Swayamdipta, and Thomas Wolf. 2019. Transfer learning in natural language processing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*, pages 15–18.

Mrinmaya Sachan and Eric Xing. 2018. Self-training for jointly learning to ask and answer questions. In *North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pages 629–640.

Kuniaki Saito, Donghyun Kim, Stan Sclaroff, and Kate Saenko. 2020. Universal domain adaptation through self supervision. *Advances in Neural Information Processing Systems*, 33.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Association for Computational Linguistics (ACL)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Duyu Tang, Nan Duan, Zhao Yan, Zhirui Zhang, Yibo Sun, Shujie Liu, Yuanhua Lv, and Ming Zhou. 2018. Learning to collaborate for question answering and asking. In *North American Chapter of the Association for Computational Linguistics(NAACL-HLT)*, pages 1564–1574.

Nicola Ueffing. 2006. Self-training for machine translation. In *NIPS workshop on Machine Learning for Multilingual Information Access*.

Jihan Yang, Shaoshuai Shi, Zhe Wang, Hongsheng Li, and Xiaojuan Qi. 2021. St3d: Self-training for unsupervised domain adaptation on 3d objectdetection. *arXiv preprint arXiv:2103.05346*.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196, Cambridge, Massachusetts, USA. Association for Computational Linguistics.

Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. 2018. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 3901–3910.

Xiaojin Zhu. 2005. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison.

A Appendix

A.1 Model Training Details

All experiments are run with same training configuration. Mean scores across 5 individual runs are provided on the test set. We describe the full model training details below for reproducibility.

BART Question Generation Transformer
We train BART-Base[2] with batch size 32 and learning rate of 1e-5. For all experiments we train the model for 5 epochs, though the model converges in 2-3 epochs. However the supervised upper-bound model (trained on MLQuestions train data starting from NQ checkpoint) is trained for 15 epochs, as the model takes 11-12 epochs to converge. For optimization we use Adam (Kingma and Ba, 2015) with $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 1e - 6$. The question and passage length is padded to 150 and 512 tokens respectively. For decoding we set beam size as 5 and remove duplicate trigrams in beam search similar to (Fan et al., 2018). The model is trained with standard cross-entropy objective. The threshold τ_G for consistency filtering is set to 1.35. The value is arrived at by plotting the log probability distribution $\log(P(q|p;\boldsymbol{\theta}))$ for $\mathrm{MLQuestions}$ dev set on initial source domain model $\boldsymbol{\theta}_G$. We consider the second quartile (Q2) as our threshold, thus 50% of the data is accepted.

Dense Passage Retriever (DPR)
We use publicly available implementation of DPR model[3] to train our IR system. The model is trained for 5 epochs with batch size of 8 for all experiments with default hyperparameter settings in (Karpukhin et al., 2020). (Karpukhin et al., 2020) also construct negative examples for each (passage, question) pair where the model maximizes question similarity with gold passage and minimizes similarity with negative passages simultaneously. We construct negative passages similar to (Karpukhin et al., 2020) as the top-k passages returned by BM25 which match most question tokens but don't contain the answer. We set $k = 7$ for our experiments. For iterative refinement models, we always use same negative passages as the model obtained after 1st iteration ($T = 1$). This is because after each iteration model is being *fine-tuned* starting from previous model and not *re-trained* on pseudo-data. We obtain better performance gains on dev set following this setting. The consistency threshold τ_R is set to 66.68. The value is determined following similar process of τ_G by plotting the DPR Score $E_P(p) \cdot E_Q(q)$ on dev set. Here we set threshold to 10%, rejecting samples having lowermost 10% scores.

[2]We use huggingface BART implementation https://huggingface.co/transformers/model_doc/bart.html

[3]https://github.com/facebookresearch/DPR