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**[1,3]

[1]Mila/McGill University   [2]Korbit Technologies

[3]Facebook CIFAR AI Chair

{devang.kulshreshtha, siva.reddy}@mila.quebec, {iulian, robert}@korbit.ai

Abstract

In this work, we introduce back-training, an alternative to self-training for unsupervised domain adaptation (UDA) from source to target domain. While self-training generates synthetic training data where natural inputs are aligned with noisy outputs, back-training results in natural outputs aligned with noisy inputs. This significantly reduces the gap between the target domain and synthetic data distribution, and reduces model overfitting to the source domain. We run UDA experiments on question generation and passage retrieval from the *Natural Questions* domain to machine learning and biomedical domains. We find that back-training vastly outperforms self-training by a mean improvement of 7.8 BLEU-4 points on generation, and 17.6% top-20 retrieval accuracy across both domains. We further propose consistency filters to remove low-quality synthetic data before training. We also release a new domain-adaptation dataset-*MLQuestions* containing 35K unaligned questions, 50K unaligned passages, and 3K aligned question-passage pairs.

1 Introduction

In domains such as education and medicine, collecting labeled data for tasks like question answering and generation requires 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 pre-trained models directly without adapting to the target domain often leads to poor generalization due to distributional shift (Zhao et al., 2019). To address this issue, these models are further trained on cheap synthetically generated labeled data by exploiting unlabeled data from target domain (Ramponi and Plank, 2020). One such popular data augmentation method for unsupervised domain adaptation (UDA) is *self-training* (Yarowsky, 1995).

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, $\cdot_u$ denotes samples from the target domain and $\hat{\cdot}$ denotes the samples generated by a supervised model trained on the source domain.

In self-training, given a pre-trained model that can perform the task of interest in a source domain and unlabeled data from the target domain, the pre-trained model is used to predict noisy labels for the target domain data. The pre-trained model is then fine-tuned on synthetic data to adapt to the new domain. To improve the quality of the synthetic data, it is also common to filter out low-confidence model predictions (Zhu, 2005).

A model fine-tuned on its own confidence predictions might suffer from confirmation bias which leads to overfitting (Yu et al., 2020). This means that the distributional gap between the target domain's true output distribution and the learned output distribution could grow wider as training proceeds. In this paper, we propose a new training protocol called *back-training* which closes this gap (the name is inspired from *back-translation* for ma-

chine translation). While self-training generates synthetic data where noisy outputs are aligned with quality inputs, back-training generates quality outputs aligned with noisy inputs. The model fine-tuned to predict real target domain outputs from noisy inputs reduces overfitting to the source domain (Vincent et al., 2008), and matches the target domain distribution more closely.

We focus on unsupervised domain adaptation (UDA) of Question Generation (QG) and Passage Retrieval (IR) from generic domains such as Wikipedia to target domains. Our target domain of interest is *machine learning*, as it is a rapidly evolving area of research. QG and IR tasks could empower student learning on MOOCs (Heilman and Smith, 2010). For example, from a passage about linear and logistic regression, an education bot could generate questions such as *what is the difference between linear and logistic regression?* to teach a student about these concepts. Moreover, IR models could help students find relevant passages for a given question (Fernández-Luna et al., 2009). In this domain, unsupervised data such as text passages and questions are easy to obtain separately rather than aligned to each other.

We also perform our main domain adaptation experiments on *biomedical* domain using PubMedQA dataset (Jin et al., 2019) to further strengthen our hypothesis.

Table 1 demonstrates the differences between self-training and back-training for QG and IR. Consider the QG 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 an IR model $P_S(p|q)$ on the source domain, then sample a question q_u from the target domain distribution $P_T(q)$. We condition the retriever on this question i.e., $P_S(p|q_u)$, 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 show that QG and IR models trained on NaturalQuestions (Kwiatkowski et al., 2019) generalize poorly to target domains, with at least 17% mean performance decline on both QG and IR tasks. 2) Although self-training improves the domain performance marginally, our back-training method outperforms self-training by a mean improvement of 7.8 BLEU-4 points on generation, and 17.6% top-20 retrieval accuracy across both target domains. 3) We further propose consistency filters to remove low-quality synthetic data before training. 4) We release *MLQuestions*: a domain adaptation dataset for the machine learning domain containing 35K unaligned questions, 50K unaligned passages, and 3K aligned question-passage pairs.

2 Background

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

2.1 Source Domain: NaturalQuestions

We use the NaturalQuestions dataset (Kwiatkowski et al., 2019) as our source domain. NaturalQuestions is an open-domain question answering dataset containing questions 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 QA, 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 as per Nielsen et al. (2008). Table 2 shows these classes and their distribution. As seen, 86% of them are descriptive questions starting with *what, who, when* and *where*. Refer to Appendix A.2 for details on dataset pre-processing and Appendix A.4 for detailed taxonomy description.

2.2 Target Domain I: Machine Learning

Our first target domain of interest is machine learning. There is no large supervised QA dataset for this domain, and it is expensive to create one since it requires domain experts. However, it is relatively cheap to collect a large number of ML articles and questions. We collect ML concepts and passages

Taxonomy	Examples (from MLQuestions)	Description (Frequent Wh-words)	Distribution (%) NaturalQuestions	MLQuestions
DESCRIPTION	What is supervised learning with example?	Asking definition or examples about a concept (What, Who, When, Where)	86%	39%
METHOD	How do you compute vectors in Word2Vec?	Computational or procedural questions - (How)	1%	15%
EXPLANATION	Why does ReLU activation work so surprisingly well?	Causal, justification or goal-oriented questions - (Why)	3%	18%
COMPARISON	What is the difference between LDA and PCA?	Ask to compare more than one concept with each other	5%	10%
PREFERENCE	Is language acquisition innate or learned?	Yes/No or select from valid set of options - (Is, Are)	5%	18%

Table 2: Classification of 200 random questions from NaturalQuestions and MLQuestions as per Nielsen (2008).

from the Wikipedia machine learning page[1] and recursively traverse its subcategories. We end up with 1.7K concepts such as *Autoencoder*, *word2vec* etc. and 50K passages related to these concepts.

For question mining, we piggy-back on Google Suggest's *People also ask* feature to collect 104K questions by using above 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 46K in-domain questions, and has 92% accuracy upon analysing 100 questions. Of these, we use 35K questions as unsupervised data. See appendix A.3 for classifier training details and performance validation.

The rest of the 11K questions are used to create supervised data for model evaluation. We use the Google search engine to find answer passages to these questions, resulting around 11K passages. Among these, we select 3K question and passage pairs as the evaluation set for QG (50% validation and 50% test). For IR, 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.

2.3 Target Domain II: Biomedical Science

Our second domain of interest is biomedicine for which we use PubMedQA (Jin et al., 2019) dataset.

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

Questions are extracted from PubMed abstract titles ending with question mark, and passages are the conclusive part of the abstract. As unsupervised data, we utilize PQA-U(nlabeled) subset containing 61.2K unaligned questions and passages. For supervised data, we use PQA-L(abeled) subset of 1K question-passage pairs manually curated by domain experts. We use the same dev-test split of 50-50% as (Jin et al., 2019) as the evaluation set for QG. For IR, in order to have the same number of candidate passages as MLQuestions, we combine randomly sampled 10K passages from PQA-U with 1K PQA-L passages to get 11K passages as candidate passages for 1K questions.

2.4 Question Generation Model

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

2.5 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. For model training details, see A.1.
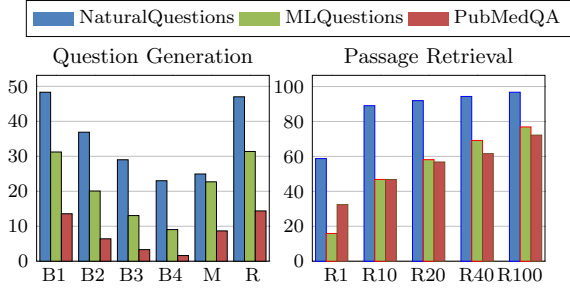
Figure 1: IID/OOD generalization gaps for Question Generation and Passage Retrieval due to distributional shift between source and target domains. For a fair comparison, the number of candidate passages for IR are kept similar for all datasets.

2.6 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). They are abbreviated as B1, B2, B3, B4, M, and R respectively throughout the paper. We also perform human evaluation on the model generated questions. For passage retrieval, we report top-k retrieval accuracy for $k = 1, 10, 20, 40, 100$ following Karpukhin et al. (2020) by measuring the fraction of cases where the correct passage lies in the top k retrieved passages. We consider 11K passages in all datasets for retrieval during test time.

3 Transfer from Source to Target Domain without Adaptation

We investigate how well models trained on NaturalQuestions transfer directly to our target domains 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 and PubMedQA for QG and IR tasks. Figure 1 shows the results. We observe high performance drops across all generation metrics (14-20%) from NaturalQuestions (IID data) to MLQuestions and PubMedQA (OOD Data). Human evaluation on QG (see Table 7) also reveals that the generated questions are either generic, or fail to understand domain-specific terminology. OOD performance in the IR task is even worse (25-40% drop), revealing a huge distribution shift between the source and target domain.

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

Table 3: Notations used throughout the paper.

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 introduce consistency filters to further improve the quality of the synthetic data.

4.1 Problem Setup

The source domain consists of labeled data containing questions paired with passages $\mathcal{D}_S \equiv \{(q_s^i, p_s^i)\}_{i=1}^m$. The target domain consists of unlabeled passages $\mathcal{P}_U \equiv \{p_u^i\}_{i=1}^{m_p}$, and unlabeled questions $\mathcal{Q}_U \equiv \{q_u^i\}_{i=1}^{m_q}$. Note that $\mathcal{P}_U$ and $\mathcal{Q}_U$ are *not necessarily* aligned with each other. Given this setup, our goal is to learn QG and 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 (Yarowsky, 1995) 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 $\mathcal{D}_S$. Then, at each iteration, the above models generate pseudo-labeled data from unlabeled passages $\mathcal{P}_U$ for question generation and questions $\mathcal{Q}_U$ for passage retrieval. For QG, $\boldsymbol{\theta}_G$ generates a question $\hat{q}$ for each $p_u \in \mathcal{P}_U$ and adds $(p_u, \hat{q})$ to synthetic data S_G. For IR, $\boldsymbol{\theta}_R$ retrieves a passage $\hat{p}$ from $\mathcal{P}_U$ for each $q_u \in \mathcal{Q}_U$ 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. The process is repeated for a desired number of iterations, which we refer to as *iterative refinement*. Note that in self-training, inputs are sampled from 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

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

---

Require: Source Data $\mathcal{D}_\mathcal{S} \equiv \{(q_s^i, p_s^i)\}_{i=1}^m$, Target unlabeled data $\mathcal{P}_\mathcal{U} \equiv \{p_u^i\}_{i=1}^{m^p}$, $\mathcal{Q}_\mathcal{U} \equiv \{q_u^i\}_{i=1}^{m_q}$
Ensure: Target domain QG model $\boldsymbol{\theta_G}$, IR model $\boldsymbol{\theta_R}$
1: **Init:** $\boldsymbol{\theta_G}, \boldsymbol{\theta_R} \leftarrow$ Train on $\mathcal{D}_\mathcal{S}$
2: **repeat**
3:     $S_G \leftarrow [\,], S_R \leftarrow [\,]$ ▷ Synthetic data for $\boldsymbol{\theta_G}$ and $\boldsymbol{\theta_R}$
4:     **for** $q_u \in \mathcal{Q}_\mathcal{U}$ **do**
5:         $\hat{p} \leftarrow$ Retrieve p from $\mathcal{P}_\mathcal{U}$ closest to q_u using $\boldsymbol{\theta_R}$
6:         add $(\hat{p}, q_u)$ to S_R S_G
7:     **end for**
8:     **for** $p_u \in \mathcal{P}_\mathcal{U}$ **do**
9:         $\hat{q} \leftarrow$ Generate q from p_u using $\boldsymbol{\theta_G}$
10:         add $(p_u, \hat{q})$ to S_G S_R
11:     **end for**
12:     $\boldsymbol{\theta_G} \leftarrow$ Finetune on S_G, $\boldsymbol{\theta_R} \leftarrow$ Finetune on S_R
13: **until** *dev performance decreases*

Figure 2: Self-training and Back-training for UDA.

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 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. 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 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 *full* unlabeled data along with their predictions even if the predictions are of low confide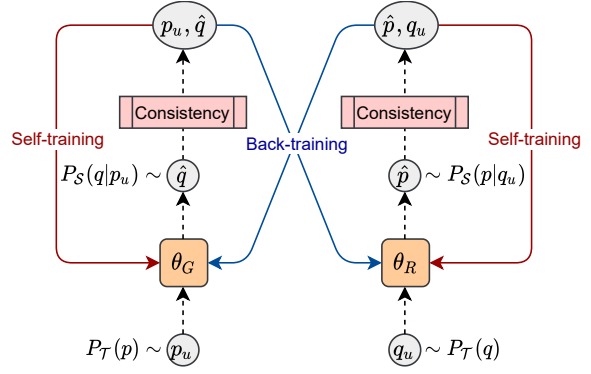nce. To alleviate this problem, in self-training, it is common to filter low-confidence predictions (Zhu, 2005). We generalize this notion as *consistency filtering*: For the tasks QG and IR, a *generator* $G \in \{\boldsymbol{\theta_G}, \boldsymbol{\theta_R}\}$ produces synthetic training data for a 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** where the generator and critic are the *same*. This is equivalent to filtering out model's own low confidence predictions in self-training. 2) **Cross consistency** where the generator and critic are *different*. This means $\boldsymbol{\theta_R}$ will filter the synthetic data generated by $\boldsymbol{\theta_G}$, and vice-versa. For $\boldsymbol{\theta_G}$ as critic we use conditional log-likelihood $\log Pr(q|p; \boldsymbol{\theta_G})$ as the confidence score. For $\boldsymbol{\theta_R}$ as critic we use the dot product similarity between the encodings $E_P(p)$ and $E_Q(q)$ as the confidence score. Self-training and back-training can be combined with one or both of the these consistency checks. We set filter thresholds to accept 75% of synthetic data (refer to appendix A.1 for exact threshold values).

A popular data filtering technique in data augmentation is cycle consistency (Alberti et al., 2019) which is enforced by further generating noisy input from noisy output, and matching noisy input similarity with source input. We leave its exploration as future work.

5 Domain Adaptation Evaluation

As described in Section 2, our source domain is NaturalQuestions and the target domains are MLQuestions and PubMedQA. We evaluate if domain adaptation helps to improve the performance compared to no adaptation. We empirically investigate qualitative differences between self-training and back-training to validate their effectiveness. We also investigate if consistency filters and iterative refine-

Dataset	Model	Question Generation						Passage Retrieval			
		B1	B2	B3	B4	M	R	R@1	R@20	R@40	R@100
MLQuestions	No-adaptation	31.23	20.07	13.05	9.04	22.70	31.38	15.86	58.13	69.13	76.86
	Self-Training	31.81	20.74	13.61	9.43	23.31	32.18	17.86	65.26	74.13	83.06
	Back-Training	**44.12**	**32.86**	**24.21**	**18.48**	**23.83**	**43.97**	**24.53**	**77.73**	**84.8**	**91.73**
PubMedQA	No-adaptation	13.57	6.41	3.31	1.62	8.67	14.38	32.4	56.8	61.6	72.2
	Self-Training	13.36	6.28	3.25	1.64	8.84	15.00	32.8	57.0	63.6	72.8
	Back-Training	**26.71**	**17.01**	**11.80**	**8.25**	**16.99**	**25.14**	**55.4**	**79.8**	**81.8**	**85.8**

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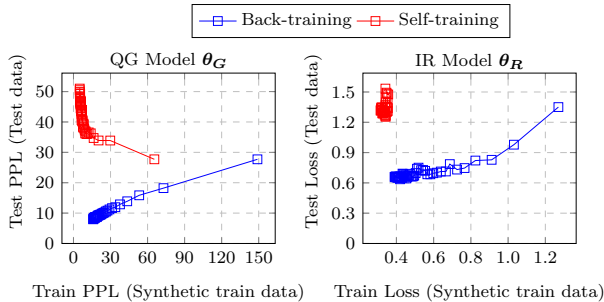

Figure 3: Evolution of QG model perplexity (PPL) and IR model loss for Self-training vs Back-training as training proceeds on MLQuestions. *Trajectories run from right to left as training loss decreases with time.* Rightmost points are plotted after first mini-batch training, and subsequent points are plotted after each mini-batch training.

Figure 4: DPR embedding similarity scores and QG Log-likelihood scores distribution on MLQuestions synthetic data computed using θ_R and θ_G respectively.

ment 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 (i.e., without consistency filtering or iterative refinement) with the no-adaptation baseline (i.e. model trained on source domain and directly tested on target domain). On MLQuestions, self-training achieves an absolute gain of around 0.6 BLEU-4 points for QG and 7.13 R@20 points for IR. Whereas back-training vastly outperforms self-training, with improvements of 9.4 BLEU-4 points on QG and 19.6 R@20 points on IR over the no-adaptation baseline. The improvements are even bigger on PubMedQA whereas self-training shows no improvement at all.

5.2 Why does back-training work?

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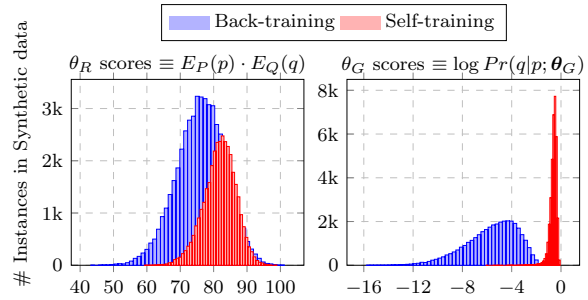
loss (and hence likelihood) are correlated and hence the data generated by back-training matches the target distribution more closely than self-training; (2) self-training achieves lower training error but higher test error compared to back-training, indicating overfitting; (3) extrapolating back-training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

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

5.3 Are consistency filters useful?

Table 5 reveals that although our consistency filters outperform base models on MLQuestions, the improvements are not very significant. Our hypothesis is that quality of synthetic data is already high (as backed up by Section 5.2 findings), which limits the performance gain. However, the filters reduce synthetic training data by 25%, which leads to faster model training without any drop in performance. Additionally, self-consistency improves

Consistency	QG		IR	
	BLEU4	ROUGE	R@20	R@100
Self-Training				
None	9.43	32.18	65.26	83.06
Self	**9.85**	**32.34**	64.75	**83.23**
Cross	8.92	31.97	**65.46**	83.00
Back-Training				
None	18.48	43.97	77.73	91.73
Self	18.62	**44.19**	77.40	91.66
Cross	18.67	43.22	**78.86**	**92.13**

Table 5: Effect of using consistency filters on Self-Training and Back-Training for MLQuestions.

Iteration	QG		IR	
	BLEU4	ROUGE	R@20	R@100
Self-Training				
$T = 1$	9.43	32.18	65.26	83.06
$T = 2$	9.28↓	32.09↓	65.60↑	83.78↑
Net Gain	0	0	0.34	0.72
Back-Training				
$T = 1$	18.48	43.97	77.73	91.73
$T = 2$	20.01↑	46.02↑	79.80↑	93.26↑
Net Gain	1.53	2.05	2.07	1.53

Table 6: Evolution of model performance on MLQuestions with increasing iterations: Blue numbers denote increases in performance, while Red numbers denote decrease in performance.

self-training in many problems (Zhu, 2005; Sachan and Xing, 2018). We believe our cross-consistency filter could also be explored on similar problems.

5.4 Is iterative refinement useful?

Further performance improvement of up to 1.53 BLEU-4 points and 2.07 R@20 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 MLQuestions 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. From the results in Table 7, we observe that the

back-training model is superior on all four criteria. However, all models perform similarly on *naturalness*.

In Table 8 we present some generated questions of various models on MLQuestions and PubMedQA dataset. Subjectively, we find that no-adaptation and self-training models fail to understand domain knowledge, generate generic questions and miss important words present in gold question. Whereas back-training generated question matches more closely to gold question.

Model	N	C	FC	A
No-adaptation	0.64	0.30	0.58	0.68
Self-Training	0.63	0.32	0.58	0.70
Back-Training	**0.66**	**0.41**	**0.64**	**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).

5.6 Analysis of Question Types

	D	C	E	M	P
D	79	7.7	4.2	6.7	2.2
C	18	74.5	4.2	2.4	0.8
E	47.8	11.8	29.6	8.4	1.7
M	28.7	5.3	1.2	64	0.6
P	53.5	12.5	5.3	7.1	21.4

Figure 5: Confusion matrix of actual (row) vs model generated question (column) classes for 100 questions sampled from MLQuestions test set. Classes are abbreviated as Description (D) Comparison (C), Explanation (E), Method (M), and Preference (P). Values are in % where each row sums to 100%.

We analyze how well our QG model can generate different kinds of questions according to the taxonomy described in Table 2. In Figure 5 we plot the confusion matrix between the actual question class and generated question class for our back-training model. To do this, 100 actual questions

Passage	Questions
If the line is a good fit for the data then the residual plot will be random. However, if the line is a bad fit for the data then the plot of residuals will be random.	**No-adaptation:** What is the meaning of random plot in statistics? **ST:** What is the meaning of random plot in statistics? **BT:** How do you know if a residual plot is random? **Reference:** How do you know if a residual plot is good?
Financial incentives for smoking cessation in pregnancy are highly cost-effective, with an incremental cost per quality adjusted life years of £482, which is well below recommended decision thresholds.	**No-adaptation:** When do we stop smoking in pregnancy? **ST:** When do you stop smoking in pregnancy? **BT:** Is there a financial incentive for smoking cessation in pregrancy? **Reference:** Are financial incentives cost-effective to support smoking and cessation during pregnancy?

Table 8: Examples of generated questions from MLQuestions (first row) and PubMedQA (second row). ST and BT refer to Self-training and Back-training models respectively.

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 methods have focused on training neural Seq2Seq models (Du et al., 2017; Mishra et al., 2020; Zhao et al., 2018; Chan and Fan, 2019; Klein and Nabi, 2019) on supervised QA datasets such as SQuAD (Rajpurkar et al., 2016). Many recent works such as (Tang et al., 2017; Wang et al., 2017) recognize the duality between QG and QA and propose joint training for the two. Duan et al. (2017) generate QA pairs from YahooAnswers, and improve QA by adding a question-consistency loss in addition to QA loss. Our work instead establishes strong duality between QG and IR task. Ours is also the first work towards unsupervised domain adaptation for QG to the best of our knowledge.

Passage Retrieval has previously been performed using classical Lucene-BM25 systems (Robertson and Zaragoza, 2009) based on sparse vector representations of question and passage, and matching keywords efficiently using TF-IDF. Recently, Karpukhin et al. (2020) show that fine-tuning dense representations of questions and passages on BERT outperforms classical methods by a strong margin. We adopt the same model for domain adaptation of IR. Concurrent to our work, Reddy et al. (2020) also perform domain adaptation for IR. Our focus has been on systematically approaching UDA problem for both QG and IR.

Data Augmentation methods like self-training have been applied in numerous NLP problems such as 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 question generation and question answering (QA) in the same domain, and filter data using QA model confidence on answer generated by question.

Back-translation's idea of aligning real outputs with noisy inputs is shared with back-training and has been successful in improving Unsupervised NMT (Artetxe et al., 2018; Edunov et al., 2018). Zhang et al. (2018) use back-translation to generate synthetic data for the task of automatic style transfer. Back-training also shares similarities with co-training (Blum and Mitchell, 1998; Wan, 2009) and tri-training (Li et al., 2014; Weiss et al., 2015) where multiple models of *same* task generate synthetic data for each other.

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 by a large margin on our newly released dataset MLQuestions and PubMedQA.

One area of future research will be exploring back-training for other paired tasks like visual question generation (Mostafazadeh et al., 2016) and image retrieval (Datta et al., 2008), and style transfer (Gatys et al., 2015) from source to target domain and vice-versa. The theoretical foundations for the superior performance of back-training have to be explored further.

Acknowledgments

References

Chris Alberti, Daniel Andor, Emily Pitler, Jacob Devlin, and Michael Collins. 2019. Synthetic QA corpora generation with roundtrip consistency. In *ACL*.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. Unsupervised statistical machine translation. In *EMNLP*.

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*.

Yoshua Bengio. 2012. Deep learning of representations for unsupervised and transfer learning. In *ICML workshop on unsupervised and transfer learning*.

Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *CoNLL*.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *SemEval*.

Ying-Hong Chan and Yao-Chung Fan. 2019. Bert for question generation. In *INLG*.

Yu-An Chung, Hung-Yi Lee, and James Glass. 2018. Supervised and unsupervised transfer learning for question answering. In *NAACL*.

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)*.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *ACL*.

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *EMNLP*.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *EMNLP*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *ACL*.

Juan M Fernández-Luna, Juan F Huete, Andrew Mac-Farlane, and Efthimis N Efthimiadis. 2009. Teaching and learning in information retrieval. *Information Retrieval*.

Leon A Gatys, Alexander S Ecker, and Matthias Bethge. 2015. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*.

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

Michael Heilman and Noah A Smith. 2010. Good question! statistical ranking for question generation. In *NAACL*.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. 2019. Pubmedqa: A dataset for biomedical research question answering. In *EMNLP*.

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 *EMNLP*.

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

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. *TACL*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pretraining for natural language generation, translation, and comprehension. In *ACL*.

Zhenghua Li, Min Zhang, and Wenliang Chen. 2014. Ambiguity-aware ensemble training for semi-supervised dependency parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 457–467, Baltimore, Maryland. Association for Computational Linguistics.

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

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*.

Mary L McHugh. 2012. Interrater reliability: the kappa statistic. *Biochemia medica*.

Robert J Menner. 1936. The conflict of homonyms in english. *Language*.

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 *ACL*.

R Nielsen. 2008. Question generation: Proposed challenge tasks and their evaluation. In *Workshop on the Question Generation Shared Task and Evaluation Challenge, Arlington, Virginia*.

Rodney D Nielsen, Jason Buckingham, Gary Knoll, Ben Marsh, and Leysia Palen. 2008. A taxonomy of questions for question generation. In *Workshop on the Question Generation Shared Task and Evaluation Challenge*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*.

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

Alan Ramponi and Barbara Plank. 2020. Neural unsupervised domain adaptation in NLP—A survey. In *COLING*.

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 *NAACL: Tutorials*.

Mrinmaya Sachan and Eric Xing. 2018. Self-training for jointly learning to ask and answer questions. In *NAACL*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *ACL*.

Duyu Tang, Nan Duan, Tao Qin, Zhao Yan, and Ming Zhou. 2017. Question answering and question generation as dual tasks. *arXiv preprint arXiv:1706.02027*.

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

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *ICML*.

Xiaojun Wan. 2009. Co-training for cross-lingual sentiment classification. In *ACL*.

Tong Wang, Xingdi Yuan, and Adam Trischler. 2017. A joint model for question answering and question generation. *arXiv preprint arXiv:1706.01450*.

David Weiss, Chris Alberti, Michael Collins, and Slav Petrov. 2015. Structured training for neural network transition-based parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 323–333, Beijing, China. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *NAACL*.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *ACL*.

Xiang Yu, Ngoc Thang Vu, and Jonas Kuhn. 2020. Ensemble self-training for low-resource languages: grapheme-to-phoneme conversion and morphological inflection. In *SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*.

Zhirui Zhang, Shuo Ren, Shujie Liu, Jianyong Wang, Peng Chen, Mu Li, Ming Zhou, and Enhong Chen. 2018. Style transfer as unsupervised machine translation. *arXiv preprint arXiv:1808.07894*.

Han Zhao, Remi Tachet Des Combes, Kun Zhang, and Geoffrey Gordon. 2019. On learning invariant representations for domain adaptation. In *ICML*.

Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. 2018. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *EMNLP*.

Xiaojin Zhu. 2005. Semi-supervised learning literature survey. Technical report, 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. For optimization we use Adam (Kingma and Ba, 2015) with $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 1e-8$. The question and passage length is padded to 150 and 512 tokens respectively. For decoding we use top-k sampling (Fan et al., 2018) with $k = 50$. The model is trained with standard cross-entropy objective.

Dense Passage Retriever (DPR)

We use publicly available implementation of DPR model[3] to train our IR system. We also use pre-trained NQ DPR checkpoint provided by authors [4] as the model trained on source domain of NaturalQuestions dataset. The model is trained for 5 epochs with batch size of 32 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.

---

[2]We use huggingface BART implementation https://huggingface.co/transformers/model_doc/bart.html
[3]https://github.com/facebookresearch/DPR
[4]https://github.com/facebookresearch/DPR/blob/master/dpr/data/download_data.py

Consistency	Critic	
	$\boldsymbol{\theta}_G$	$\boldsymbol{\theta}_R$
Self consistency	-1.19	78.24
Cross consistency	-5.95	71.65

Table 9: Threshold values for different consistency filters. Values are chosen as the third quartile (Q3) of score distribution of synthetic data, accepting 75% of synthetic data for model training.

Figure 6: Test set Confusion matrix of Out-of-domain (OOD) and In-domain classes for classifier probability threshold of 0.8.

Consistency Filters

Table 9 enlists threshold values for different consistency filters. Values are arrived at by plotting confidence scores distribution of synthetic data, and setting threshold to accept 75% of the data (i.e. third quartile Q3). As explained in section 4.4, for $\boldsymbol{\theta}_G$ as critic we use conditional log-likelihood $\log Pr(q|p; \boldsymbol{\theta}_G)$ as our confidence scores. For $\boldsymbol{\theta}_R$ as critic we use DPR similarity score $E_P(p)E_Q(q)$ as our confidence scores.

A.2 NaturalQuestions Dataset Pre-processing

We use Google NaturalQuestions dataset as our *source* domain corpus. We pre-process publicly available train and dev corpora in a similar manner to (Mishra et al., 2020) by selecting all questions starting from the *long-answer* tag and filtering out cases where the long-answer doesn't start with the HTML <p> tag. We obtain 108,501 examples which we split into a 90/10 ratio for training/dev sets. The NQ dev set of 2,136 examples is used as our test data (as the test set is hidden).
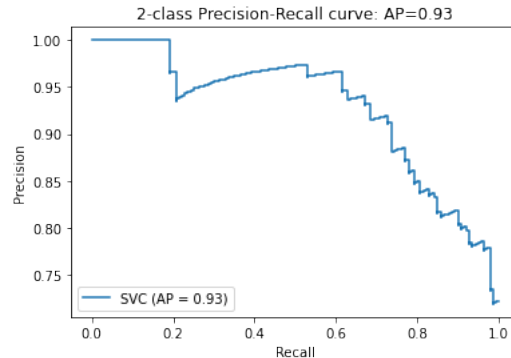

Figure 7: Precision-Recall curve for Test set of 150 questions. AP denotes average precision.

A.3 MLQuestions: Filtering undesirable data

This section describes filtering out-of-domain questions (OOD) from collected 104K questions from Google described in section 2.2. Many ML terms are *homonyms* (Menner, 1936): they have a different meaning in another context - (e.g. "Ensemble", "Eager Learning", "Transformers"). This means the collected data contains OOD questions. Upon analyzing 100 random questions drawn from 104K questions, we find 27 of them are OOD.

To filter such undesirable data, we randomly sample 1000 questions and recruit 3 domain experts to label them as In-domain or OOD. 200 questions were labeled by all 3 to determine inter-annotator agreement. We record a Cohen's Kappa agreement score (McHugh, 2012) of 0.84. The 1000 annotated questions are split into sizes 800, 50, 150 for train, dev, and test sets respectively. Based on this labeled data, we train a classifier on top of question features to classify remaining questions as *useful* or *OOD*. For extracting features from questions, we utilize DistillBERT model (Sanh et al., 2019) trained on SNLI+MultiNLI (Bowman et al., 2015; Williams et al., 2018) and then fine-tuned on the STS benchmark(Cer et al., 2017) train set[5]. This gives us feature vector of size 768 which is used to train SVM classifier[6] with L2 penalty of 0.1. We carefully set the acceptance threshold relatively high to 0.8, to ensure high precision, thus accepting very few OOD questions.

Figure 6 shows confusion matrix on test set with α set as 0.8. The classifier obtains high precision and average recall of 94.6% and 66% respectively.

---

[5] We use off-the-shelf implementation `https://github.com/UKPLab/sentence-transformers` to extract sentence features from pretrained model

[6] `https://scikit-learn.org/stable/modules/svm.html`

High precision is empirically verified by annotating 100 random accepted questions, out of which 92 are found to be in-domain. The remaining 8% of the data can be treated as noise for model training. Figure 7 plots the precision-recall trade-off by varying the acceptance threshold α.

A.4 Taxonomy of MLQuestions Dataset

In Table 2, we show the distribution of various types of questions in MLQuestions and NaturalQuestions dataset. We split the questions into 5 categories based on Nielsen's Educational Taxonomy (Nielsen et al., 2008): *descriptive* questions, which ask for definitions or examples; *method* questions which ask for computations or procedures; *explanation* questions, which ask for justifications; *comparison* questions, which ask to compare two or more concepts; and *preference* questions, which are answered by a selection from a set of options. Refer to Nielsen (2008) for detailed understanding of the taxonomy.

B Reproducibility Checklist

B.1 For all reported experimental results

- *A clear description of the mathematical setting, algorithm, and/or model:* This is provided in Section 2 and Appendix A.1 of the main paper.

- *Submission of a zip file containing source code, with specification of all dependencies, including external libraries, or a link to such resources (while still anonymized):* We provide the source code zipped repository *MLQuestions*. The README file contains all instructions needed to replicate experiments. The file *requirements.txt* specifies required python dependencies.

- *Description of computing infrastructure used:* We perform our experiments on a machine with specifications: 2 CPUs, 2 RTX8000 GPUs, 18GB RAM.

- *The average runtime for each model or algorithm (e.g., training, inference, etc.), or estimated energy cost:* Table 10 lists average runtime for each step of vanilla Self-training and back-training algorithms, as well as for consistency filters.

- *Number of parameters in each model:* For question generation, the BART base model

Task	Data Size	Runtime
θ_G generates synthetic data	50K	174
θ_R generates synthetic data	35K	110
θ_G filters low-quality data	35K	31
θ_G filters low-quality data	50K	45
θ_G filters low-quality data	35K	35
θ_G filters low-quality data	50K	48
θ_G Self-training	50K	373
θ_G Back-training	35K	263
θ_R Self-training	35K	547
θ_R Back-training	50K	762

Table 10: Runtime (in minutes) for each step in domain adaptation models for MLQuestions dataset. Since there are 35K unaligned questions and 50K unaligned passages, a step has different execution times depending on type of training (self/back) or consistency filter (self/cross).

contains total 139M parameters. For passage retrieval, the DPR model contains total 220M parameters.

- *Corresponding validation performance for each reported test result:* Tables 11, 12, 13 report the validation set performance for each reported test result in the main paper.

- *Explanation of evaluation metrics used, with links to code:* Refer to Section 2.6 of main paper for explanation of evaluation metrics. For evaluating QG model, we use the Maluuba NLG-Eval github library to compute BLEU, METEOR, ROUGE scores. The repository can be found at `https://github.com/Maluuba/nlg-eval`. For IR, we implement the top-K retrieval accuracy which can be found in the file location `file://MLQuestions/IR/eval_retriever.py` of our submitted source code.

B.2 For all experiments with hyperparameter search

- *The exact number of training and evaluation runs:* For all experiments we train the QG and IR for 5 epochs. We evaluate the model performance using evaluation metrics after each epoch on the validation set, and find that models converge after 2-3 epochs.

- *Bounds for each hyperparameter:* We experimented by manually varying hyperparameters in vicinity of values mentioned in appendix A.1. The best hyperparameters on val-

Dataset	Model	Question Generation						Passage Retrieval			
		B1	B2	B3	B4	M	R	R@1	R@20	R@40	R@100
MLQuestions	No-adaptation	30.64	19.70	12.82	8.80	23.23	31.33	13.00	54.86	64.6	73.93
	Self-Training	31.01	20.36	13.50	9.37	23.67	31.75	14.13	62.20	70.80	80.66
	Back-Training	**41.42**	**30.72**	**22.50**	**17.29**	**23.38**	**41.58**	**21.86**	**77.40**	**84.66**	**90.26**
PubMedQA	No-adaptation	14.23	7.02	3.65	1.81	9.12	15.96	32.66	57.20	61.8	72.68
	Self-Training	14.04	6.98	3.09	1.54	8.67	15.30	33.0	57.48	64.2	73.44
	Back-Training	**27.17**	**17.92**	**12.34**	**8.76**	**17.66**	**25.89**	**56.60**	**81.0**	**83.20**	**87.68**

Table 11: Validation set results of unsupervised domain adaptation. *No-adaptation* denotes the model trained on NaturalQuestions and evaluated directly on MLQuestions/PubMedQA dev sets without any domain adaptation.

	QG		IR	
Consistency	BLEU4	ROUGE	R@20	R@100
	Self-Training			
None	9.37	31.75	62.20	80.66
Self	**9.76**	31.67	62.75	81.56
Cross	9.02	**32.34**	**62.96**	**82.00**
	Back-Training			
None	17.29	41.58	77.40	90.26
Self	17.91	**43.27**	76.86	91.06
Cross	**18.09**	41.84	**78.26**	**91.33**

Table 12: Effect of using consistency filters on Self-Training and Back-Training for MLQuestions validation set.

	QG		IR	
Iteration	BLEU4	ROUGE	R@20	R@100
	Self-Training			
$T=1$	9.37	31.75	62.20	80.66
$T=2$	9.22↓	31.13↓	62.80↑	81.08↑
Net Gain	0	0	0.60	0.42
	Back-Training			
$T=1$	17.29	41.58	77.40	90.26
$T=2$	19.97↑	45.74↑	78.56↑	91.26↑
Net Gain	2.68	4.16	1.16	1.00

Table 13: Evolution of model performance on MLQuestions validation set with increasing iterations: Blue numbers denote increases in performance, while Red numbers denote decrease in performance.

idation set were chosen for final model training.

- *Hyperparameter configurations for best-performing models:* We provide complete hyperparameters details for QG and IR model in Appendix A.1.

- *The method of choosing hyperparameter values (e.g., uniform sampling, manual tuning, etc.) and the criterion used to select among them (e.g., accuracy):* We use manual tuning method with the criterion as BLEU-4 accuracy for QG and R@40 retrieval accuracy for IR task on validation set.

- *Summary statistics of the results (e.g., mean, variance, error bars, etc.):* Mean scores across 5 individual runs are provided for all experiments of main paper.

B.3 For all datasets used

- *Relevant details such as languages, and number of examples and label distributions:* Section 2 provide statistics of NaturalQuestions, MLQuestions, and PubMedQA datasets. All datasets are in English language.

- *Details of train/validation/test splits:* This is also provided in section 2 for all three datasets.

- *Explanation of any data that were excluded, and all pre-processing steps:* Relevant details are provided in section 2 for all three datasets.

- *A zip file containing data or link to a downloadable version of the data:* We provide MLQuestions dataset in the submission zip file. The NaturalQuestions and PubMedQA dataset can be downloaded from https://ai.google.com/research/NaturalQuestions/download and

https://github.com/pubmedqa/
pubmedqa repsectively. The datasets can
be pre-processed following the procedures
mentioned in section 2.

- *For new data collected, a complete descrip-
tion of the data collection process, such as
instructions to annotators and methods for
quality control.:* We provide above details for
our newly created dataset *MLQuestions* in sec-
tion 2.2.