Unsupervised Adaptation of Question Answering Systems via Generative Self-training

Steven J. Rennie, Etienne Marcheret, Neil Mallinar, David Nahamoo, and Vaibhava Goel
Pryon Inc., Brooklyn, New York, New York.
{srennie, emarcheret, nmallinar, dnahamoo, vgoel}@pryon.com

Abstract

BERT-era question answering systems have recently achieved impressive performance on several question-answering (QA) tasks. These systems are based on representations that have been pre-trained on self-supervised tasks such as word masking and sentence entailment, using massive amounts of data. Nevertheless, additional pre-training closer to the end-task, such as training on synthetic QA pairs, has been shown to improve performance. While recent work has considered augmenting labelled data and leveraging large unlabelled datasets to generate synthetic QA data, directly adapting to target data has received little attention. In this paper we investigate the iterative generation of synthetic QA pairs as a way to realize unsupervised self adaptation. Motivated by the success of the roundtrip consistency method for filtering generated QA pairs, we present iterative generalizations of the approach, which maximize an approximation of a lower bound on the probability of the adaptation data. By adapting on synthetic QA pairs generated on the target data, our method is able to improve QA systems significantly, using an order of magnitude less synthetic data and training computation than existing augmentation approaches.

1 Introduction

Supervised self-training methods have transformed applied machine learning recently. Such tasks serve as "pre-training" for related downstream tasks, and have proven to be essential to attaining state-of-the-art performance, particularly in NLP.

BERT-era Transformer-based question answering systems have recently achieved impressive performance on several question-answering (QA) tasks. These systems are based on representations that have been pre-trained on self-supervised tasks such as word masking and sentence entailment, using massive amounts of data (Devlin et al., 2018; Liu et al., 2019; Yang et al., 2019; Dong et al., 2019; Radford et al., 2019). Nevertheless, additional pre-training closer to the end-task, such as training on synthetic QA pairs, has been shown to improve performance (Alberti et al., 2019; Dong et al., 2019). While recent work has considered augmenting labelled data and leveraging large unlabelled datasets to generate synthetic QA data, directly adapting to target data has, to our knowledge, not been investigated in the context of BERT-era modeling and performance levels.

Recently, roundtrip consistency (RTC) was introduced as a criteria for filtering synthetic question-answer pairs on unlabelled data, and has demonstrated solid gains when applied to large unlabelled datasets to generate millions of RTC-validated pairs as *task-specific* pre-training data (Alberti et al., 2019). Such *unsupervised* self-training can be an effective way to de-emphasize low confidence predictions, adapt to the target input distribution, distill decoding procedures, and instill input response invariances.

In this paper we present new theoretical justification for RTC, and explore novel *iterative* generalizations of RTC for adapting in a *task-specific*, *target data specific* manner. We show that most of these approaches optimize an approximation of a lower bound of the probability of the data, and thereby can, beyond self-training, potentially also adapt to explain the target data more effectively. Under the formulation, the question-answering system is used as a surrogate likelihood function for the question and answer generators. In this manner, the difficult task of modeling the generation of entire contexts is avoided: instead abstractive parts of the context (the answers), whose locations are latent and estimated, are utilized, similar to an autoencoder, but with questions as the latent code,

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 1148–1157,
November 16–20, 2020.

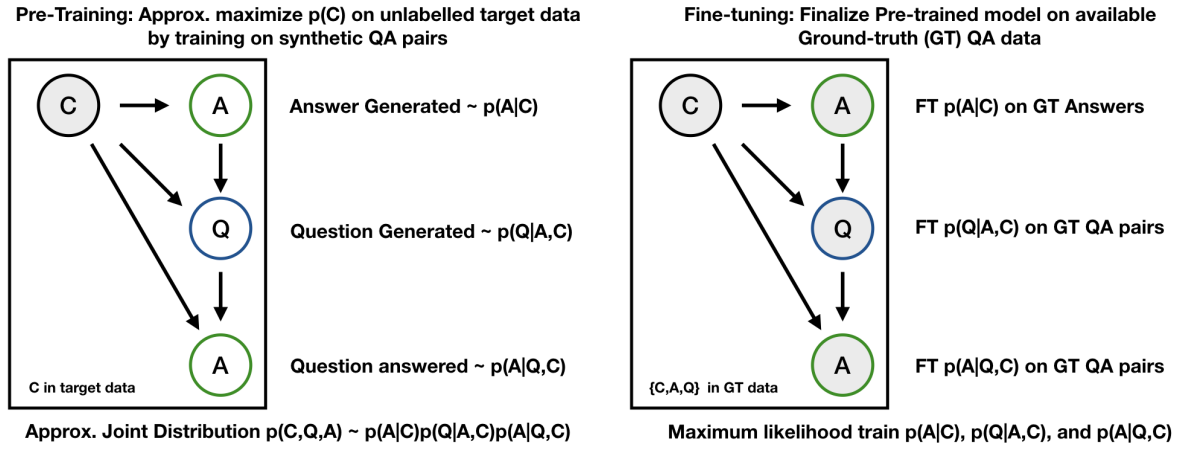

Figure 1: Overview of Approach: We adapt to unlabelled target data by iterating between generating synthetic question answer (QA) pairs and approximately maximizing the probability of the observed contexts C, and fine-tuning on all available ground truth (GT) QA data. Please refer to (6-9) for further details.

and latent answer inputs. By adapting on synthetic QA pairs generated on the target data, our method is able to improve QA systems significantly, using an order of magnitude less synthetic data and training computation than existing augmentation approaches.

The main contributions of this paper are as follows:

- We consider the problem of adapting QA systems to target data using synthetically generated QA pairs, and show that this improves QA systems significantly, while using an order of magnitude less data and computation relative to existing methods that augment using large unlabelled data sets.

- We present a solid theoretical foundation for understanding existing and developing new synthetic data filtering algorithms, including the effective but elusive Roundtrip Consistency (RTC) algorithm.

- We compare several related methods for utilizing synthetic data to adapt to target data, by iteratively pre-training and fine-tuning the QA system and QA generators, and show that some variations optimize an approximation of a lower bound on the probability of the adaptation data, despite being composed on only inference networks.

2 Supervised Training of QA systems

Typically question-answering (QA) systems are trained to produce an abstractive answer A, given a question Q, and the relevant local context C.

The QA model is generally trained to maximize the probability of QA pairs in context:

$$l_{a|g} = \sum_t \log p(A_t^* | Q_t^*, C_t^*) \quad (1)$$

Where $\{Q_t, A_t, C_t\}$ denotes a set of human-annotated question-answer-context triples (Rajpurkar et al., 2016). Supervised training of Question-Answering systems is effective, but ground-truth (GT) QA pairs are cumbersome and expensive to annotate. Bootstrapping from pre-trained representations such as BERT significantly reduces the number of GT QA pairs that are required to train a high quality QA system, but current systems still currently require hundreds of thousands of GT QA pairs on data matching the style and content of the target data to perform at state-of-the-art levels.

3 Generating Synthetic QA Data

To augment an existing QA training set, several authors have used the supervised data to train question generators (Sachan and Xing, 2018; Duan et al., 2017; Alberti et al., 2019; Dong et al., 2019). As with the QA counterparts, these models are typically trained in a supervised manner:

$$l_{q|a} = \sum_t \log p(Q_t^* | A_t^*, C_t^*) \quad (2)$$

and then employed to augment the training set with additional, synthetic questions, for each ground-truth answer.

Most techniques that operate fully unsupervised also train an answer prior in the same manner, which generates answers for a given context, which the question generator then conditions on to generate questions (Alberti et al., 2019):

$$l_a = \sum_t \log p(A_t^*|C_t^*) \quad (3)$$

4 Iterative Pre-train then Fine-tune Based Adaptation

Most existing methods for synthetic data augmentation utilize QA generators trained only on supervised data. This synthetic data is used either as pre-training data for the QA system, which is then fine-tuned on ground-truth data (Alberti et al., 2019), or just simply as additional GT data for training.

In this paper we investigate iteratively pre-training both the QA system and the QA generators on the synthetic data generated by the most recent fine-tuned QA generators. The iterative procedure is as follows:

1. Generate QA pairs using the current fine-tuned QA generators.
2. Pre-train the QA generators and QA system on the (filtered) generated data from scratch.
3. Fine-tune the QA generators and QA system on ground truth data.
4. Repeat until converged or for a maximum number of iterations.

5 Fundamentals of Unsupervised Self-Training

Training a model on it's own predictions, or *pseudo-truth*, has a long history in machine learning and speech and language applications (Scudder, 1965; Novotney and Callison-Burch, 2010). It generally leads to performance improvements, but why? The expected gradient of the log probability of a model $p(y|x)$ trained on its own predictions is zero: $E_{y\sim p_{\hat{\theta}}(y|x)}[\nabla_\theta \log p_\theta(y|x)] = 0$! There are few ways that self-training can improve performance:

- By re-prioritizing the model's capacity based on a target input distribution x that is generated for, i.e., by compensating for *covariate shift* (Sugiyama and Kawanabe, 2012).
- By sharpening the model to eliminate prediction noise, via the distillation of any decoding processes (e.g. beam search) (Grandvalet and Bengio, 2005).
- By introducing perturbations that the model's predictions are trained to be invariant to.

Recently in (He et al., 2019), it was shown that for machine translation, significant gains could be realized via self-training, and that most of this gain could be attributed to the last effect: in particular, invariance to input and dropout noise. In summary, predictive models can benefit substantially from self-training: one does not need to improve a generative model of the data to improve prediction performance on that data. Nevertheless, we will next relate the predictive models used in QA generators and QA systems to underlying implicit generative models, to derive insight and justify existing QA filtering methods, and derive new ones. It turns out that several self-training algorithms for QA optimization actually optimize an approximate lower bound on the probability of the data, and so have the potential to evolve their representations to maximize the probability of the target data.

6 A Generative Framework for Adapting QA Systems

In this section we consider a simple generative framework for understanding predictive QA systems. In unsupervised data settings, only the context C is observed, and the questions $\{Q\}$ and answers $\{A\}$ are latent, and must be inferred.

Here we consider the following generative model for each context:

$$\log p(C) = \log \sum_{Q,A} p(C|Q,A)p(Q|A)p(A) \quad (4)$$

Where the index of the context in the adaptation data has been omitted to avoid notational clutter. Typically the factors of a generative models are explicit, and each factor is endowed with parameters, to maximize the probability of the data. In this paper, these factors, as we shall see, are only implicit, and we explore the possibility of maximizing the probability of the data, using *only* inference networks, which condition on the context: namely the answer generator, $p(A|C)$, the question generator, $p(Q|A,C)$, and the QA system, $p(A|Q,C)$. Our interest in doing so is 3 fold: 1) To avoid having to specify and train a generative model of contexts,

which may be difficult to learn. 2) To better understand the theoretical underpinnings of existing augmentation algorithms such as the roundtrip consistency algorithm (RTC) (Alberti et al., 2019), and 3) Foremost, to achieve the end-goal of deriving insight that leads to more effective iterative algorithms for adapting QA systems.

We begin our quest by low-bounding the probability of the data, with a posterior distribution $q(A, Q)$ over the latent questions and answers for the context, as is done for VAES (Kingma and Welling, 2013).

$$\log p(C) \geq \sum_{Q,A} q(Q, A|C) \log \frac{p(C|Q, A)p(Q, A)}{q(Q, A|C)} \tag{5}$$

Expanding the prior term $p(Q, A)$ in terms of question and answer generators, $p(Q|A, C)$ and $p(A|C)$, we have:

$$\begin{aligned} \log p(Q, A) &= \log \sum_{C'} p(Q|A, C')p(A|C')\hat{p}(C') \\ &\geq \log p(Q|A, C)p(A|C)\hat{p}(C) \end{aligned} \tag{6}$$

Where $\hat{p}(C')$ is the current estimate of probability of context C'. In general we expect this lower bound to be quite strong, as $p(Q|A, C')$ and $p(A|C')$ are identically or close to zero for most $C' \neq C$. The resulting lower bound is on $p(C)$ is:

$$\log p(C) \geq \mathcal{L} = \log \hat{p}(C) + \sum_{Q,A} q(Q, A|C) \log \frac{p(C|Q, A)p(Q|A, C)p(A|C)}{q(Q, A|C)} \tag{7}$$

Which can be maximized to improve $p(C)$. The resulting bound looks much like a typical variational bound (e.g. 5), the principle difference being that the distributions being optimized are themselves inference networks, and condition on C. The effective joint distribution corresponding to this bound is:

$$p(C, Q, A) \approx p(C|Q, A)p(Q|A, C)p(A|C) \tag{8}$$

7 Defining the inference distribution q

In contrast with how VAEs are typically optimized, here we utilize multiple samples to form a posterior over QA pairs (Burda et al., 2015; Rainforth et al., 2018), and employ a form of prioritized truncated variational inference (TVI) to combat mode-dropping, which was recently shown to be a proper variational bound that can be iterated (Lücke, 2016). TVI in short, utilizes a subset Φ of joint states to define the posterior, i.e. $q(Q, A) \propto p(Q, A, C), \{Q, A\} \in \Phi, 0\, o.w$, but any subset of states, and any convex set of weights, can be used to form a lower bound.

$$\begin{aligned} \log p(C) &= \log \sum_{Q,A} p(C, Q, A) \\ &\geq \log \sum_{Q,A \in \Phi} p(C, Q, A) \\ &\geq \sum_{Q,A \in \Phi} q(Q, A|C) \log \frac{p(C, Q, A)}{q(Q, A|C)} = \mathcal{L}_\Phi(C) \end{aligned} \tag{9}$$

When $q(Q, A|C) \propto p(Q, A, C)$, $\{Q, A\} \in \Phi, 0\, o.w$, the second bound is tight. In general $q(Q, A|C)$ takes the form:

$$q(Q, A|C) = \frac{q(Q, A, C)}{q(C)} \tag{10}$$

where:

$$q(C) = \sum_{Q,A \in \Phi} q(Q, A, C) \tag{11}$$

The consequence of this is that all the bounds derived above hold, even if the sums over all QA pairs are taken over a prioritized set. The process of defining q thus can be decomposed into two steps:

1. define the subset of QA pairs to include in the set Φ
2. define the convex weights on the identified set, $q(Q, A|C)$.

8 Training Objectives

Typically when the end goal is data modeling, the probability of each data example is maximized indiscriminately. The maximum likelihood objective over the adaptation dataset $\{C_t\}$ in this case is:

$$O_{ML} = \sum_t \mathcal{L}_\Phi(C_t) \tag{12}$$

However, when training inference networks to make predictions for a given set of contexts in an unsupervised manner, it is natural to put more emphasis on confident predictions. A natural self-training objective in this context is to weigh each

context according to it's current estimated probability, so as to avoid re-enforcing poor predictions:

$$O_{ST(\alpha)} = \sum_t q(C_t)^\alpha \mathcal{L}_\Phi(C_t) \quad (13)$$

With $\alpha \geq 0$. When $\alpha = 0$, The standard ML objective is recovered. When $\alpha = 1$, the normalization and the overall objective reduces to self-training on the current estimate of the joint distribution over $\{Q, A, C\}$ triples:

$$O_{ST(1)} = \sum_t \sum_{\{Q,A\} \in \Phi_t} q(Q, A, C_t) \log p(Q, A, C_t) \quad (14)$$

We found that $O_{ST(1)}$ consistently, leads to slightly better QA performance than pre-training with O_{ML}, and so we optimize $O_{ST(1)}$ during the pre-training step for all results in this paper, except where otherwise noted.

9 The question answer posterior (QAP) based likelihood approximation

As discussed above, summing over all QA pairs is intractable, and so we select the set Φ by prioritizing wrt $p(Q|A, C)p(A|C)$ via beam search. Given the set, we need to approximate $p(Q, A, C)$. Here we approximate the likelihood function $p(C|Q, A)$ by the QA posterior $p(A|Q, C)$, which will attribute low scores to poor QA pairs. Since the answers A are abstractive, this is a good approximation. With $p(C|Q, A) \approx p(A|Q, C)$, and Φ prioritizing wrt $p(Q|A, C)p(A|C)$ via beam search, we utilize the tightest possible TVI bound on (7):

$$q(Q, A, C) \propto p(A|Q, C)p(Q|A, C)p(A|C) \quad (15)$$

Treating these prioritized samples as independent samples from $p(Q|A, C)p(A|C)$ [1], the final weight on each sample after importance sampling is proportional to $p(A|Q, C)$. The resulting per example adaptation objective for our generators and QA system are given by:

$$O = \sum_{Q,A \in \Phi} \tilde{p}(A|Q, C)[\log p_{\theta_{a|q}}(A|Q, C) + \log p_{\theta_{q|a}}(Q|A, C) + \log p_{\theta_a}(A|C)] \quad (16)$$

Where $\tilde{p}(A|Q, C) = \frac{p(A|Q,C)}{(\sum_{Q,A \in \Phi} p(A|Q,C))^\alpha}$ when the objective $O_{ST(\alpha)}$ is used ($p(A|Q, C)$ when $\alpha = 1$).

In essence, the generators are trained on the reward signal $\tilde{p}(A|Q, C)$. Note that the expected gradient of the QA term is not zero, because the set set has been prioritized. In addition, dropout is used during training, to encourage invariance to dropout noise on *generated* inputs. Adding noise to the input would similarly improve performance, but we leave this to future work. Effectively this approach further prioritizes the set of questions produced by beam search based on the posterior distribution of the answer that the question was generated for, and encourages prediction consistency over noisy representations.

10 Interpreting existing approaches

Current techniques can be understood in the context of the above bounds, which provides new theoretical justification for these methods, and furthermore justifies the iterative extensions of these methods presented herein.

10.1 Beam Search and Augment (BSA)

The most straightforward approach to incorporating synthetic data is to generate QA pairs using beam search using the generators, $p(A|C), p(Q|A, C)$, and then treating the generated data as ground-truth data with weight 1. Selecting a subset Φ_{QA} of QA pairs here corresponds to an instance of truncated variational inference to define the support of $q(Q, A)$, and using weights of one corresponds to a uniform distribution over the selected set, for lack of any way to approximate the likelihood function, and use of $O_{ST(\alpha)}$ with $\alpha = 1$.

10.2 Roundtrip Consistency

Roundtrip consistency, introduced in (Alberti et al., 2019), generates a set of candidate QA pairs using beam search, and accepts the QA pair only if $\arg\max_A p(A|Q_i, C) = A_i$. The authors sketch two potential avenues for the formal justification of RTC in the appendix, but state that "a key question for future work is to develop a more formal understanding of why the roundtrip method improves accuracy".

Under the presented framework, RTC can be easily interpreted. The RTC procedure for validating QA pairs is an instance of the Iterated Conditional Modes (ICM) algorithm (Besag, 1986) for identifying local modes of a posterior distribution. The

[1] In practice prioritized samples often lead to better performance when treated as independent samples (Alberti et al., 2019), we also found that this was the case.

ICM algorithm consists of iteratively maximizing the conditional posterior of one variable (set), given all the others, until convergence. The resulting set of variables is a "conditional mode": the posterior mode of each variable (set), conditioned on all other variables are consistent. RTC performs only one iteration, and discards the example if consistency is not satisfied. In this manner RTC goes further than beam search, and accepts only prioritized generations that are conditional modes of the underlying posterior as members of the set Φ_{QA}. Like BSA, accepted QA pairs are given weight 1, and so like BSA, corresponds to utilizing a uniform distribution over the (smaller) set Φ that satisfies cycle consistency, and the self-training objective $O_{ST(\alpha)}$ (14) with $\alpha = 1$.

The QAP approach to approximating the likelihood represents a soft generalization of RTC, where QA pairs are re-weighted based on the *probabilistic* cycle consistency criterion $q(Q, A) \propto p(A|C)p(Q|A, C)p(A)$, which, as we have shown, produces the tightest bounds on $\log p(C)$ under the model for *any* prioritized set Φ. To retain the speed and performance advantages of selecting only modes, but still assign soft scores, we set a threshold of 0.5 on the QAP filter, so that only modes will be selected, but they will have soft scores associated with them.

11 Related Work

Several authors have recently investigated the use of synthetic data to improve question answering systems, with most of the work we are aware of either augmenting the training set, or generating synthetic QA pairs on auxiliary data (Sachan and Xing, 2018; Du et al., 2017; Duan et al., 2017; Song et al., 2017; Alberti et al., 2019; Dong et al., 2019; Zhang and Bansal, 2019). In most existing work, the generators are trained only on ground-truth data. One exception is (Sachan and Xing, 2018), which jointly self-trains question given answer (RNN) and answer given question (Attentive Reader) models. Another is (Wang et al., 2019), which introduces a domain classifier to adapt to target domain data. However, in contrast with our work, they train their QA system component only on ground-truth source data, because they found that synthetic data degrades QA system performance, whereas our PT QA systems trained *only* on synthetic data often outperform the baseline.

In this work, we focus on using synthetic data to adapt to target data, and iteratively improve both the generators and the QA system during the adaptation process using pre-trained Transformers, by alternating between self-supervised pre-training and ground-truth fine-tuning phases. The closest existing work to ours that we are aware of is the Roundtrip Consistency paper itself (Alberti et al., 2019), whose "fine-tuning only" experiments parallel ours, in that they fine-tune pre-trained BERT models to define the answer and question generators, and the QA system (SQUAD2 gain 0.9 F1 with 3M QA pairs). Note that, in contrast with our work, their generators were trained only on ground-truth data, and the augmentation process was not iterated.

An important element of this work is to show that iterative Roundtrip Consistency filtering and the Answer Posterior filtering method proposed here approximately optimize a lower bound on the probability of the target domain data. In (Lewis and Fan, 2018), the authors propose a generative model for question answering, which purposefully avoids the use of a QA decoder, $p(A|Q, C)$, instead inverting the generative model $p(A|C), p(Q|A, C)$ explicitly over a prioritized set of answers using beam search, so that an answer must be able to generate the question to be an answer. Additional discriminative training is required to make the approach work well, but nevertheless, this desiderata is captured by both Roundtrip Consistency filtering and Answer Posterior filtering: yes, generated questions are verified by virtue of matching generated and predicted answers, but also answers are validated by the generators' ability to communicate it when the answer$\rightarrow$ question $\rightarrow$ answer loop is enforced. Yet another manifestation of this idea is regularizing the question generator $p(Q|A, C)$ and question answering system $p(A|Q, C)$ to encourage them to be consistent during (supervised) training (Tang et al., 2017; Duan et al., 2017).

In (Dong et al., 2019) the authors investigate the use of UniLM on SQUAD and fine-tune UniLM as a question generator to augment the training set with new questions for ground truth answers. Our preliminary experiments indicate that this is a straightforward way to significantly boost baseline QA and Q generator performance before adapting to target data, given that the ground truth answers are available, using the methods described herein.

Method	QAP selection	QAP weight
BSA	$TOPK[p(A\|C)p(Q\|A,C)]$	1
RTC	$TOPK[p(A\|C)p(Q\|A,C)]$ & $\arg\max_A p(A\|Q_i,C) = A_i$	1
QAP	$TOPK[p(A\|C)p(Q\|A,C)]$ & $p(A_i\|Q_i,C) > 0.5$	$p(A_i\|Q_i,C)$

Table 1: Summary of iterative adaptation techniques investigated in this paper: Beam Search Adaptation (BSA), Roundtrip Consistency (RTC), and Question Answer Posterior (QAP). Composition of the approximate QAC likelihood, which defines the weight of each generated example during pre-training, consists of 1) Selecting a set Φ of QA pairs for the context C, and 2) setting a weight for each selected pair.

12 Experiments

12.1 Test Scenarios

To evaluate the efficacy of unsupervised self-training for QA system adaptation, we consider the following scenarios:

- Well Matched Target Conditions, Limited Training Data (WM-LT)
- Well Matched Target Conditions, Plentiful Training Data (WM-PT)
- Mismatched Target Conditions, Plentiful Training Data (MM-PT)

In this paper well matched conditions (WM-LT, WM-PT) are assessed by adapting SQUAD-trained models on SQUAD development data (Rajpurkar et al., 2016), and mismatched conditions by adapting SQUAD-trained models on the portion of the Natural-Questions (NQ) development data (Kwiatkowski et al., 2019) that contains abstractive short answers.

12.2 Models

In this paper all QA generation and answering models are trained by fine-tuning BERT (base, uncased, unless noted otherwise). The question generator is trained as a left-to-right sequence-to-sequence model, which conditions on the observed context. Ground-truth or generated answers are marked in the context by introducing additional segment ids to mark the answer. All models are iteratively pre-trained on synthetic generated data and fine-tuned on SQUAD using essentially the standard BERT fine-tuning recipe for SQUAD (ADAM optimizer, std. parameters; lr=3e-5; 2 epochs, "warmup linear" schedule for both PT and FT). All experiments were conducted on 4 or 8 node V-100 machines, and all models were adapted for two iterations.

Model	EM	F1
Baseline-FT	–/69.6	–/79.7
BAS-PT	56.7/58.1	69.0/71.3
BAS-FT	71.1/71.1	81.2/81.2
RTC-PT	65.6/67.4	75.4/77.5
RTC-FT	71.6/71.6	81.2/81.5
QAP-PT	64.7/66.1	75.2/77.2
QAP-FT	71.5/71.7	81.4/81.8

Table 2: Limited training data scenario: Squad 1.1 QA adaptation EM/F1 results on dev (GT data is 9K of Squad1.1 train, bsize 24, pt bsize 80, 4/8 q/a per c,first/best iter. results shown). See section 12.3 for details, and table 1 for a description of each technique.

12.3 Results: Well Matched, Limited Training Data

Table 2 depicts SQUAD 1.1 QA adaptation results on dev (GT data is 9K of SQUAD 1.1 training set, ft batch size 24, pre-training batch size 80). For this test the answer nbest and question nbest were set to 8 and 4 respectively, yielding 32 QA pairs per paragraph (2554 dev paragraphs) before filtering. The performance of the first and best iteration are depicted for each metric, for all models. Looking at the results, we can see that RTC-FT and QAP-FT, which do additional filtering/re-weighting based on QA feedback, respectively, slightly outperform using the prioritized beam search results as gt adaptation data (BSA-FT). Note however, that both RTC and QAP are generally significantly more efficient to train than BSA, as a significant percentage of the prioritized synthetic data is discarded (the amount discarded depends on both the strength of the generator and how mismatched the target conditions are, see table 5 for further analysis). The performance gap between pre-trained models (PT), which are trained only on generated data, in contrast, is more marked. Filtering improves PT model performance substantially.

Model	EM	F1
Baseline	–/46.8	–/62.6
BSA-PT	34.4/24.4	51.2/51.2
BSA-FT	47.0/47.0	62.8/62.8
RTC-PT	42.9/43.8	58.0/58.7
RTC-FT	47.4/47.8	63.1/63.3
QAP-PT	43.3/44.4	57.8/58.3
QAP-FT	48.1/48.1	63.7/63.7
Baseline (LMPT)	–/47.6	–/63.4
QAP-PT (LMPT)	42.9/44.7	57.2/59.1
QAP-FT (LMPT)	48.2/48.5	63.9/64.1

Table 3: Mismatched Target Data Scenario: NQ results on dev (GT data is all of the Squad1.1 training set, bsize 24, pt bsize 96, 4/16 q/a per c, first/best iter. results shown). See section 12.4 and table 1 for details.

12.4 Results: Mismatched Target Conditions, Plentiful Training Data

Table 3 depicts SQUAD 1.1 QA adaptation results on dev (GT data is all, $\sim$ 90K, of the Squad1.1 training set, ft batch size 24, pre-training batch size 96). For this test the answer nbest and question nbest were set to 16 and 4 respectively, yielding 64 QA pairs per paragraph (3703 dev paragraphs) before filtering. Here we also compare QAP-based adaptation to and in combination with BERT-based bidirectional LM-pretraining (LMPT), and find that 1) QAP outperforms LMPT, and 2) LMPT and then QAP adaptation applied in succession leads to the best results. Iterative QAP significantly outperforms the baseline, and slightly outperforms both iterative RTC and BSA, but all results are significantly lower that on the SQUAD dev set. An important difference between SQUAD and NQ is that NQ answers are often significantly longer, which is a prominent bias in the definition of what an answer is, which may be difficult to overcome with just unsupervised adaptation. Nevertheless, all 3 methods are able to improve the baseline system somewhat, and outperform and improve upon LMPT.

12.5 Results: Well Matched Target Conditions, Plentiful Training Data

Table 4 depicts SQUAD 1.1 QA adaptation results on dev (GT data is 9K of Squad1.1 training set, ft batch size 24, pre-training batch size 80 unless otherwise noted). For this test the answer nbest and question nbest were set to 8 and 4 respectively, yielding 32 QA pairs per paragraph (2554 dev paragraphs) before filtering. Here all filtering/reweighting schemes perform similarly and lead to gains, despite the well matched adaptation data scenario. Increasing the number of answers and questions/answer to 8 and 32 results in additional gain, improving baseline F1 performance from 88.4 $\rightarrow$ 89.5 with QAP. We also found that not adapting the answer prior $p(A|C)$ and mean-normalizing the data weights further improved performance, particularly for QAP (c.f. Table 5, starred results).

12.6 Performance vs. # Synthesized QA Pairs

Synthetic generation and pre-training time scale linearly with the number of generated QA pairs (the latter dominating, due to backpropagation), but nevertheless, our per-paragraph adaptation levels can be significantly increased to yield significant performance gains. Table 5 compares the performance of RTC and QAP in limited training data conditions. The performance gain moving from 4/8 Q/A to 8/32 Q/A per context paragraph is substantial, and remarkably, the PT models, trained only on generated data, outperform the baseline, thanks largely in part to invariances learned with dropout on during self-training. Figure 2 depicts performance of QAP as a function of iteration and synthesis level in the well-matched, limited training data (WM-LT) condition.

12.7 Speed

Both RTC and QAP generally are significantly faster than basic BSA while performing on-par or better than basic beam search augmentation (7, 6 hrs vs 26 hrs during the first iteration using 8 V-100s, for 32 a., 8 q/a per c, see Table 5). Qualitatively we found that BSA generally doesn't benefit from additional iterations, while the performance of RTC and QAP generally saturates after 2-3 iterations, and delivers consistently better performance with lower total training time.

13 Discussion and Future Work

In this paper we have related self-training of question answering systems to the generative modeling of their associated context, and showed the former can be specified to optimize an approximate lower bound on the probability of the data. We then investigated iterative "pre-train then fine-tune" approaches to target domain adaptation, proposed question answer posterior (QAP) as an alternative form of consistency filtering, and provided theoreti-

Model	EM	F1	A, Q/A
Baseline	–/80.9	–/88.4	4, 8
BSA-PT	66.1/66.1	77.9/77.9	4, 8
BSA-FT	81.6/81.6	89.1/89.1	4, 8
QAP-PT	70.1/70.7	80.5/81.2	4, 8
QAP-FT	81.4/81.4	89.0/89.0	4, 8
QAP-PT*	77.7/78.3	85.5/86.1	8, 32
QAP-FT*	82.1/82.3	89.3/89.5	8, 32

Table 4: Well Matched Target Data Scenario: Squad 1.1 results on dev (GT data is all of the Squad1.1 training set, bsize 24, pt bsize 32, 4/8 q/a per c, first/best iter. results shown). See section 12.5 and table 1 for details. *pt bsize 96; $p(A|C)$ not adapted; 8,32 a,q/a; 3 iterations.

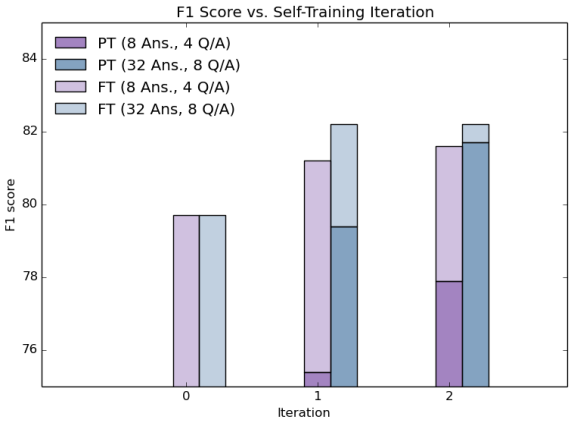

Figure 2: QAP performance as a function of iteration and # QA pairs (WM-LT condition). PT models are trained only on generated data.

Model	EM	F1	A, Q/A	Synth. Data	Relative Speed
Baseline-FT	– / 69.6	– / 79.7	–	–	–
18K Model (2X GTD)	– / 72.3	– / 81.7	–	–	–
BSA-PT/FT	58.8 / 71.9	73.6 / 82.1	32, 8	+1.1M, +1.1M	1X, 1X
RTC-PT/FT	71.6 / 72.6	81.1 / 82.4	32, 8	+220K, +360K	5X, 3X
QAP-PT/FT	72.5 / 72.7	81.7 / 82.2	32, 8	+180K, +320K	6X, 3X
RTC-PT/FT*	71.7 / 72.5	81.4/ 82.3	32, 8	+(220, 350, 422)K	(5,3,4)X
QAP-PT/FT*	72.9 / 73.4	82.2 / 83.0	32, 8	+(180,310,385)K	(6,3,3)X
BSA-PT/FT	58.1 / 71.1	71.3 / 81.2	8, 4	+130K, +130K	1X, 1X
RTC-PT/FT	67.4 / 71.6	77.5 / 81.5	8, 4	+60K, +90K	2X, 1.5X
QAP-PT/FT	68.3 / 71.8	78.0 / 81.6	8, 4	+50K, +80K	2.5X, 1.5X

Table 5: Final PT/FT performance as a function of number of prioritized questions and answers per target data context paragraph. Limited training data scenario: Squad 1.1 QA adaptation EM/F1 results on dev (GT data is 9K of Squad1.1 train). See table 1 for a description of each technique. Remarkably, the PT models, which are trained *only* on generated data, are able to outperform the baseline system, provided a sufficient number of synthetic example per target context paragraph are generated. Each PT iteration with BSA 8,32 takes 13 hrs with an 8 V100s: RTC 8,32 is 5X, and 3X faster during iteration 1&2, respectively, due to data pruning, and QAP 8, 32 is faster than RTC. All 8,32 adapted models outperform an 18K GT only model, trained on 2X the data. When we mean normalize the average training weight, do not adapt the answer prior $p(A|C)$, and adapt for 3 iterations, RTC does not improve, but QAP is significantly more effective (*).

cal justification for Roundtrip Consistency filtering. In general, the techniques work quite well, particularly in better-matched conditions. While effective, iteratively re-training the QA generators and QA system is inefficient, even with strong data filters. How to most efficiently and effectively adapt Transformer-based QA systems remains an important topic for future research.

References

Chris Alberti, Daniel Andor, Emily Pitler, Jacob Devlin, and Michael Collins. 2019. Synthetic qa corpora generation with roundtrip consistency. *arXiv preprint arXiv:1906.05416*.

Julian Besag. 1986. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society: Series B (Methodological)*, 48(3):259–279.

Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. 2015. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep

bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

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

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. *arXiv preprint arXiv:1705.00106*.

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 866–874.

Yves Grandvalet and Yoshua Bengio. 2005. Semi-supervised learning by entropy minimization. In *Advances in neural information processing systems*, pages 529–536.

Junxian He, Jiatao Gu, Jiajun Shen, and Marc'Aurelio Ranzato. 2019. Revisiting self-training for neural sequence generation. In *International Conference on Learning Representations*.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

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*, 7:453–466.

Mike Lewis and Angela Fan. 2018. Generative question answering: Learning to answer the whole question.

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*.

Jörg Lücke. 2016. Truncated variational expectation maximization. *arXiv preprint arXiv:1610.03113*.

Scott Novotney and Chris Callison-Burch. 2010. Cheap, fast and good enough: Automatic speech recognition with non-expert transcription. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 207–215. Association for Computational Linguistics.

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

Tom Rainforth, Adam R Kosiorek, Tuan Anh Le, Chris J Maddison, Maximilian Igl, Frank Wood, and Yee Whye Teh. 2018. Tighter variational bounds are not necessarily better. *arXiv preprint arXiv:1802.04537*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Mrinmaya Sachan and Eric Xing. 2018. Self-training for jointly learning to ask and answer questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 629–640.

H Scudder. 1965. Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory*, 11(3):363–371.

Linfeng Song, Zhiguo Wang, and Wael Hamza. 2017. A unified query-based generative model for question generation and question answering. *arXiv preprint arXiv:1709.01058*.

Masashi Sugiyama and Motoaki Kawanabe. 2012. *Machine learning in non-stationary environments: Introduction to covariate shift adaptation*. MIT press.

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*.

Huazheng Wang, Zhe Gan, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, and Hongning Wang. 2019. Adversarial domain adaptation for machine reading comprehension. *arXiv preprint arXiv:1908.09209*.

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

Shiyue Zhang and Mohit Bansal. 2019. Addressing semantic drift in question generation for semi-supervised question answering. *arXiv preprint arXiv:1909.06356*.