Simulating Bandit Learning from User Feedback for Extractive Question Answering

Ge Gao[♢], **Eunsol Choi**[♣] and **Yoav Artzi**[♢]
[♢]Department of Computer Science and Cornell Tech, Cornell University
[♣]Department of Computer Science, The University of Texas at Austin
ggao@cs.cornell.edu eunsol@utexas.edu yoav@cs.cornell.edu

Abstract

We study learning from user feedback for extractive question answering by simulating feedback using supervised data. We cast the problem as contextual bandit learning, and analyze the characteristics of several learning scenarios with focus on reducing data annotation. We show that systems initially trained on a small number of examples can dramatically improve given feedback from users on model-predicted answers, and that one can use existing datasets to deploy systems in new domains without any annotation, but instead improving the system on-the-fly via user feedback.

1 Introduction

Explicit feedback from users of NLP systems can be used to continually improve system performance. For example, a user posing a question to a question-answering (QA) system can mark if a predicted phrase is a valid answer given the context from which it was extracted. However, the dominant paradigm in NLP separates model training from deployment, leaving models static following learning and throughout interaction with users. This approach misses opportunities for learning during system usage, which beside several exceptions we discuss in Section 8 is understudied in NLP. In this paper, we study the potential of learning from explicit user feedback for extractive QA through simulation studies.

Extractive QA is a popular testbed for language reasoning, with rich prior work on datasets (e.g., Rajpurkar et al., 2016), task design (Yang et al., 2018; Choi et al., 2018), and model architecture development (Seo et al., 2017; Yu et al., 2018). Learning from interaction with users remains relatively understudied, even though QA is well positioned to elicit user feedback. An extracted answer can be clearly visualized within its supporting context, and a language-proficient user can then easily validate

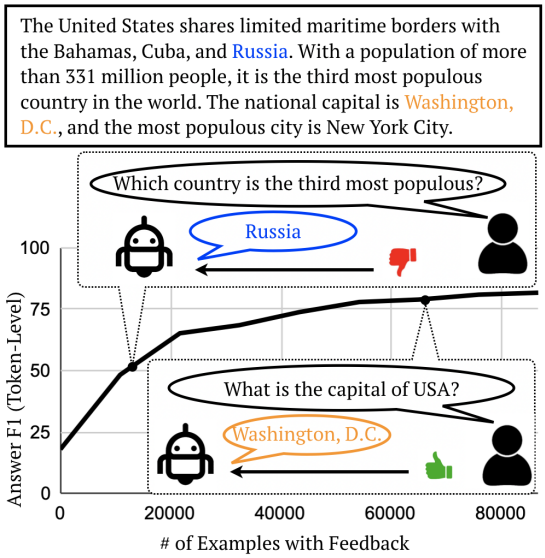

Figure 1: Illustration of an interaction setup for learning from user feedback for QA, and its potential. Given a user question, the system outputs an answer and highlights it in its context. The user validates the answer given the context with binary feedback. We show performance progression from one of our online learning experiments on SQuAD with hand-crafted illustrative examples at two time steps.

if the answer is supported or not.[1] This allows for simple binary feedback, and creates a contextual bandit learning scenario (Auer et al., 2002; Langford and Zhang, 2007). Figure 1 illustrates this learning signal and its potential.

We simulate user feedback using several widely used QA datasets, and use it as a bandit signal for learning. We study the empirical characteristics of the learning process, including its performance, sensitivity to initial system performance, and trade-offs between online and offline learning. We also simulate zero-annotation domain adaptation, where we deploy a QA system trained from supervised

[1]Answers could also come from erroneous or deceitful contexts. This important problem is not studied by most work in extractive QA, including ours. We leave it for future work.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 5167 - 5179

data in one domain and adapt it solely from user feedback in a new domain.

This learning scenario can mitigate fundamental problems in extractive QA. It reduces data collection costs, by delegating much of the learning to interaction with users. It can avoid data collection artifacts because the data comes from the actual system deployment, unlike data from an annotation effort that often involves design decisions immaterial to the system's use case. For example, sharing question- and answer-annotator roles (Rajpurkar et al., 2016), which is detrimental to emulate information seeking behavior (Choi et al., 2018). Finally, it gives systems the potential to evolve over time as the world changes (Lazaridou et al., 2021; Zhang and Choi, 2021).

Our simulation experiments show that user feedback is an effective signal to continually improve QA systems across multiple benchmarks. For example, an initial system trained with a small amount of SQuAD (Rajpurkar et al., 2016) annotations (64 examples) improves from 18 to 81.6 F1 score, and adapting a SearchQA (Dunn et al., 2017) system to SQuAD through user feedback improves it from 45 to 84 F1 score. Our study shows the impact of initial system performance, trade-offs between online and offline learning, and the impact of source domain on adaptation. These results create the base for future work that goes beyond simulation to use feedback from human users to improve extractive QA systems. Our code is publicly available at https://github.com/lil-lab/bandit-qa.

2 Learning and Interaction Scenario

We study a scenario where a QA model learns from explicit user feedback. We formulate learning as a contextual bandit problem. The input to the learner is a question-context pair, where the context paragraph contains the answer to the question. The output is a single span in the context paragraph that is the answer to the question.

Given a question-context pair, the model predicts an answer span. The user then provides feedback about the model's predicted answer, which is used to update the model parameters. We intentionally experiment with simple binary feedback and basic learning algorithms, to provide a baseline for what more advanced methods could achieve with as few assumptions as possible.

Background: Contextual Bandit Learning In a stochastic (i.i.d.) contextual bandit learning problem, at each time step t, the learner independently observes a context[2] $x^{(t)} \sim D$ sampled from the data distribution D, chooses an action $y^{(t)}$ according to a policy π, and observes a reward $r^{(t)} \in \mathbb{R}$. The learner only observes the reward $r^{(t)}$ corresponding to the chosen action $y^{(t)}$. The learner aims to minimize the cumulative regret. Intuitively, regret is the deficit suffered by the learner relative to the optimal policy up to a specific time step. Formally, the cumulative regret at time T is computed with respect to the optimal policy $\pi^* \in \arg\max_{\pi \in \Pi} \mathbb{E}_{(x,y,r)\sim(D,\pi)}[r]$:

$$R_T := \sum_{t=1}^{T} r^{*(t)} - \sum_{t=1}^{T} r^{(t)} \quad , \tag{1}$$

where Π is the set of all policies, $r^{(t)}$ is the reward observed at time t and $r^{*(t)}$ is the reward that the optimal policy π^* would observe. Minimising the cumulative regret is equivalent to maximising the total reward.[3] A key challenge in contextual bandit learning is to balance exploration and exploitation to minimize overall regret.

Scenario Formulation Let a question $\bar{q}$ be a sequence of m tokens $\langle q_1, \ldots, q_m \rangle$ and a context paragraph $\bar{c}$ be a sequence of n tokens $\langle c_1, \ldots, c_n \rangle$. An extractive QA model[4] π predicts a span $\hat{y} = \langle c_i, \ldots, c_j \rangle$ where $i, j \in [1, n]$ and $i \leq j$ in the context $\bar{c}$ as an answer. When relevant, we denote π_θ as a QA model parameterized by θ.

We formalize learning as a contextual bandit process: at each time step t, the model is given a question-context pair $(\bar{q}^{(t)}, \bar{c}^{(t)})$, predicts an answer span $\hat{y}$, and receives a reward $r^{(t)} \in \mathbb{R}$. The learner's goal is to maximize the total reward $\sum_{t=1}^{T} r^{(t)}$. This formulation reflects a setup where, given a question-context pair, the QA system interacts with a user, who validates the model-predicted answer in context, and provides feedback which is mapped to numerical reward.

[2] The term *context* here refers to the input to the learner policy, and is different from the term *context* as we use it later in extractive QA, where the term *context* refers to the evidence document given as input to the model.

[3] Equivalently, the problem is often formulated as loss minimization (Bietti et al., 2018).

[4] In bandit literature, the term *policy* is more commonly used. We use the term *model* from here on to align with the QA literature.

Algorithm 1 Online learning.

1: **for** $t = 1 \cdots$ **do**
2: Receive a question $\bar{q}^{(t)}$ and context $\bar{c}^{(k)}$
3: Predict an answer $\hat{y}^{(t)} \leftarrow \arg\max_y \pi_\theta(y \mid \bar{q}^{(t)}, \bar{c}^{(t)})$
4: Observe a reward $r^{(t)}$
5: Update the model parameters θ using the gradient
$r^{(t)} \nabla_\theta \log \pi_\theta(\hat{y}^{(t)} \mid \bar{q}^{(t)}, \bar{c}^{(t)})$
6: **end for**

Algorithm 2 Offline learning.

1: **for** $t = 1 \cdots T$ **do**
2: Receive a question $\bar{q}^{(t)}$ and context $\bar{c}^{(t)}$
3: Predict an answer $\hat{y}^{(t)} \leftarrow \arg\max_y \pi_\theta(y \mid \bar{q}^{(t)}, \bar{c}^{(t)})$
4: $p^{(t)} \leftarrow \pi_\theta(\hat{y}^{(t)} \mid \bar{q}^{(t)}, \bar{c}^{(t)})$
5: Observe a reward $r^{(t)}$
6: **end for**
7: **for** E epochs **do**
8: **for** $t = 1 \cdots T$ **do**
9: Compute clipped importance-weighted reward according to the current model parameters:
10: $r' \leftarrow \text{clip}(\frac{\pi_\theta(\hat{y}^{(t)} \mid \bar{q}^{(t)}, \bar{c}^{(t)})}{p^{(t)}}, 0, 1) r^{(t)}$
11: Update the model parameters θ using the gradient
$r' \nabla_\theta \log \pi_\theta(\hat{y}^{(t)} \mid \bar{q}^{(t)}, \bar{c}^{(t)})$
12: **end for**
13: **end for**

Learning Algorithm We learn using policy gradient. Our learner is similar to REINFORCE (Sutton and Barto, 1998; Williams, 2004), but we use $\arg\max$ to predict answers instead of Monte Carlo sampling from the model's output distribution.[5]

We study online and offline learning, also referred to as on- and off-policy. In online learning (Algorithm 1), the model identity is maintained between prediction and update; the parameter values that are updated are the same that were used to generate the output receiving reward. This entails that a reward is only used once, to update the model after observing it. In offline learning (Algorithm 2), this relation between update and prediction does not hold. The learner observes reward, often across many examples, and may use it to update the model many times, even after the parameters drifted arbitrarily far from these that generated the prediction. In practice, we observe reward for the entire length of the simulation (T steps) and then update for E epochs. The reward is re-weighted to provide an unbiased estimation using inverse propensity score (IPS; Horvitz and Thompson, 1952). We clip the debiasing coefficient to avoid amplifying examples with large coefficients (line 10, Algorithm 2).

In general, offline learning is easier to implement because updating the model is not integrated with its deployment. Offline learning also uses a training loop that is similar to optimization practices in supervised learning. This allows to iterate over the data multiple times, albeit with the same feedback signal on each example. However, online learning often has lower regret as the model is updated after each interaction. It may also lead to higher overall performance, because as the model improves early on, it may observe more positive feedback overall, which is generally more informative. We empirically study these trade-offs in Section 5 and 6.

Evaluating Performance We evaluate model performance using token-level F1 on a held-out test set, as commonly done in the QA literature (Rajpurkar et al., 2016). We also estimate the learner regret (Equation 1). Computing regret requires access to the an oracle π^*. We use human annotation as an estimate (Section 3).[6]

Comparison to Supervised Learning In supervised learning, the data distribution is not dependent on the model, but on a fixed training set $\{(\bar{q}^{(t)}, \bar{c}^{(t)}, y^{(t)})\}_{t=1}^{T}$. In contrast, bandit learners are provided with reward data that depends on the model itself: $\{(\bar{q}^{(t)}, \bar{c}^{(t)}, \hat{y}^{(t)}, r^{(t)})\}_{t=1}^{T}$ where r is the reward for the model prediction $\hat{y}^{(t)} = \arg\max_y \pi_\theta(y \mid \bar{q}^{(t)}, \bar{c}^{(t)})$ at time step t. Such feedback can be freely gathered from users interacting with the model, while building supervised datasets requires costly annotation. This learning signal can also reflect changing task properties (e.g., world changes) to allow systems to adapt, and its origin in the deployed system use makes it more robust to biases introduced during annotation.

3 Simulation Setup

We initialize our model with supervised data, and then simulate bandit feedback using supervised data annotations. Initialization is critical so the model does not return random answers, which are likely to be all bad because of the large output space. We use relatively little supervised data from the same domain for in-domain experiments (Section 5 and 6) to focus on the data annotation re-

[5] Early experiments showed that sampling is not as beneficial as $\arg\max$, potentially because of the relatively large output space of extractive QA. Yao et al. (2020) made a similar observation for semantic parsing, and Lawrence et al. (2017) used $\arg\max$ predictions for bandit learning in statistical machine translation. Table 4 in Appendix A provides our experimental results with sampling.

[6] Our oracle is an estimate because of annotation noise and ambiguity in exact span selection.

duction potential of user feedback. For domain adaptation, we assume access to a large amount of training data in the source domain, and no annotated data in the target domain (Section 7).

Reward We use supervised data annotations to simulate the reward. If the predicted answer span is an exact match index-wise to the annotated span, the learner observes a positive reward of 1.0, and a negative reward of -0.1 otherwise.[7] This reward signal is stricter than QA evaluation metrics (token-level F1 or exact match after normalization).[8]

Noise Simulation We study robustness by simulating noisy feedback via reward perturbation: randomly flipping the binary reward with a fixed probability of 8% or 20% as the noise ratio.[9]

4 Experimental Setup

Data We use six English QA datasets that provide substantial amount of annotated training data taken from the MRQA training portion (Fisch et al., 2019): SQuAD (Rajpurkar et al., 2016), NewsQA (Trischler et al., 2017), SearchQA (Dunn et al., 2017), TriviaQA (Joshi et al., 2017), HotpotQA (Yang et al., 2018), and NaturalQuestions (NQ; Kwiatkowski et al., 2019). The MRQA benchmark simplifies all datasets so that each example has a single span answer with a limited evidence document length (truncated at 800 tokens). Table 7 in Appendix B provides dataset details. We compute performance measures and learning curves on development sets following prior work (Rajpurkar et al., 2016; Ram et al., 2021).

Model We conduct experiments with a pretrained SpanBERT model (Joshi et al., 2020). We fine-tune the pre-trained SpanBERT-base model during initial learning and our simulations.

Implementation Details We use Hugging Face Transformers (Wolf et al., 2020). When training initial models with little in-domain supervised data (Section 5; Section 6), we use a learning rate of 3e-5 with a linear schedule, batch size 10, and 10 epochs. We obtain the sets of 64, 256, or 1,024 examples from prior work (Ram et al., 2021).[10] For models initially trained on complete datasets (Section 7), we use a learning rate 2e-5 with a linear schedule, batch size 40, and 4 epochs.

In simulation experiments, we use batch size 40. We turn off dropout to simulate interaction with users in deployment. For single-pass online learning experiments (Section 5; Section 7), we use a constant learning rate of 1e-5. For offline learning experiments (Section 6), we train the model for 3 epochs on the collected feedback with a linear schedule learning rate of 3e-5.

Online experiments with SQuAD, HotpotQA, NQ, and NewsQA take 2–4h each on one NVIDIA GeForce RTX 2080 Ti; 2.5–6h for offline. For TriviaQA and SearchQA, each online simulation experiment on one NVIDIA TITAN RTX takes 4–9.5h; 9–20h for offline.

5 Online Learning

We simulate a scenario where only a limited amount of supervised data is available, and the model mainly learns from explicit user feedback on predicted answers. We use 64, 256, or 1,024 in-domain annotated examples to train an initial model. This section focuses on online learning, where the learner updates the model parameters after each feedback is observed (Algorithm 1).

Figure 2 presents the performance of in-domain simulation with online learning. The performance pattern varies across different datasets. Bandit learning consistently improves performance on SQuAD, HotpotQA, and NQ across different amounts of supervised data used to train the initial model. The performance gain is larger with weaker initial models (i.e., trained on 64 supervised examples): 63.6 on SQuAD, 42.7 on HotpotQA, and 40.0 on NQ. Bandit learning is not always effective on NewsQA, TriviaQA, and SearchQA, especially with weaker initial models. This may be attributed to the quality of training set annotations, which determines the accuracy of reward in our setup. SearchQA and TriviaQA use distant supervision to match questions and relevant contexts from the web, likely decreasing reward quality in our setup. While NewsQA is crowdsourced, Trischler et al. (2017) report relatively low human performance (69.4 F1), possibly indicating data challenges that also decrease our reward quality. Learning progres-

[7] We experimented with other reward values, but did not observe a significant difference in performance (Appendix A).

[8] Normalization includes lowercasing, modifying spacing, removing articles and punctuation, etc. NaturalQuestions (NQ; Kwiatkowski et al., 2019) is an exception, with an exact index match measure that has similar strictness.

[9] Even without our noise simulation, the simulated feedback inherits the noise from the annotation, either from crowdsourcing or distant supervision.

[10] We use the seed 46 sets publicly available at https://github.com/oriram/splinter.

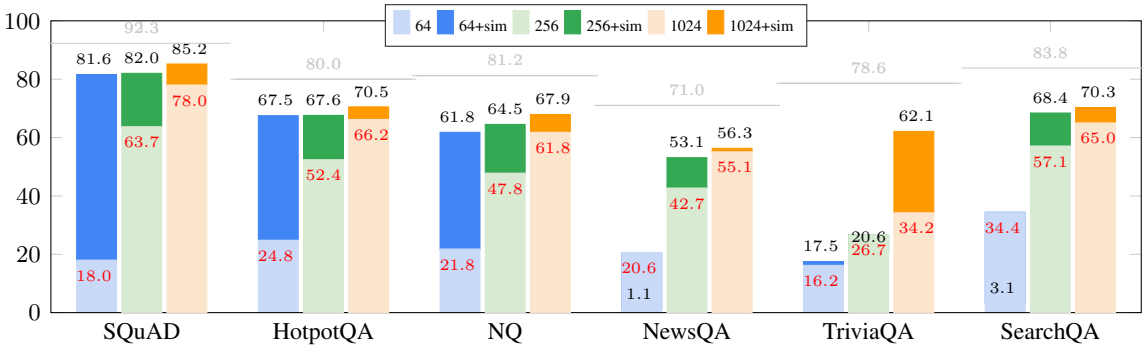

Figure 2: Online in-domain simulation development F1 performance. Horizontal grey lines represent the supervised training performance on each dataset. Data labels in red are performance of initial models trained on 64, 256, or 1024 examples (i.e., lighter bars). Darker bars and black data labels represent simulation performance. Lower simulation performance (e.g., NewsQA 64+sim) indicate degradation in performance following simulation.

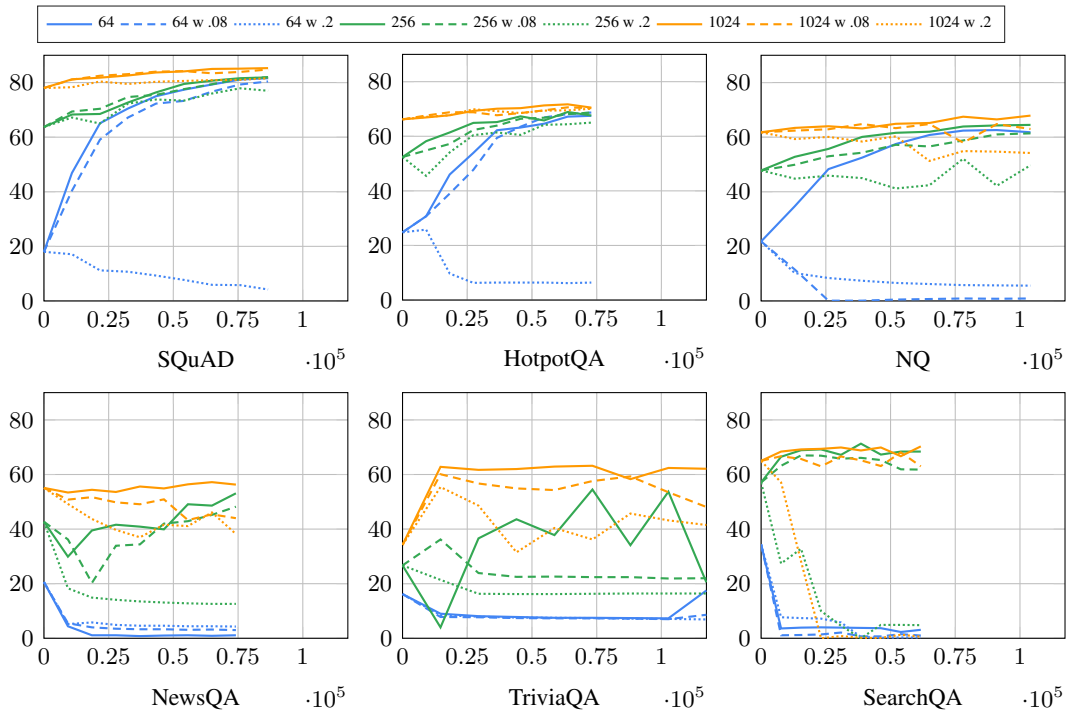

Figure 3: Online in-domain simulation development F1 learning curves. X-axis is the number of examples with feedback observed. "x w y" denotes initially training with x supervised in-domain examples and simulating with y amount of feedback noise.

Setup	SQuAD	HotpotQA	NQ	NewsQA	TriviaQA	SearchQA
64+sim	78.2(-3.4)	66.3(-1.2)	51.3(-10.5)	3.1(+2.0)	0.4(-17.1)	1.3(-1.8)
256+sim	86.2(+4.2)	70.9(+3.3)	65.2(+0.7)	54.3(+1.2)	12.3(-8.3)	0.3(-68.1)
1024+sim	86.5(+1.3)	73.2(+2.7)	71.8(+3.9)	55.7(-0.6)	7.5(-54.6)	4.1(-66.2)

Table 1: Offline in-domain simulation development F1 performance. Numbers in parenthesis show the performance gain (green) or decrease (red) of offline learning compared to online learning (Figure 2).

Setup	SQuAD	HotpotQA	NQ	NewsQA	TriviaQA	SearchQA
64+sim	0.63 / 1.04	0.51 / 0.94	0.74 / 0.91	1.07 / 0.86	0.77 / 0.77	1.09 / 0.77
256+sim	0.56 / 0.75	0.36 / 0.58	0.71 / 0.83	0.84 / 0.85	0.76 / 0.72	0.73 / 0.69
1024+sim	0.48 / 0.55	0.27 / 0.33	0.65 / 0.67	0.73 / 0.71	0.71 / 0.64	0.69 / 0.65

Table 2: Regret averaged by the number of feedback observations in online/offline in-domain simulations.

sion across datasets (Figure 3) shows that initial models trained with 1,024 examples can achieve peak performance with one third or even one quarter of feedback provided.

Feedback Noise Simulation Figure 3 shows learning curves with simulated noise via different amounts of feedback perturbation (0%, 8%, or 20%). When perturbation-free simulation is effective, models remain robust to noise: 8% noise results in small fluctuations of the learning curve, but the final performance degrades minimally. Starting with weaker initial models and learning with a higher noise ratio may cause learning to fail (e.g., simulation on SQuAD with 64 initial examples and 20% noise). When online perturbation-free simulation fails, online learning with noisy feedback fails too.

Sensitivity Analysis Training Transformer-based models has been shown to have stability issues, especially when training with limited amount of data (Zhang et al., 2021). Our non-standard training procedure (i.e., one epoch with a fixed learning rate) may further increase instability. We study the stability of the learning process using initial models trained on only 64 in-domain supervised examples on HotpotQA and TriviaQA: the former shows significant performance gain while the latter shows the opposite. We experiment with five initial models trained on different sets of 64 supervised examples, each used to initiate a separate simulation experiment. Four out of five experiments on HotpotQA show performance gains similar to what we observed so far, except one experiment that starts with very low initialization performance. In contrast, nearly all experiments on TriviaQA collapse (mean F1 of 7.3). We also conduct sensitivity analysis with stronger initial models trained with 1,024 examples, and observe that the final performance is stable across runs on both HotpotQA and TriviaQA (standard deviations are 0.5 and 2.6). Table 5 in Appendix B provides detailed performance numbers.

6 Offline Learning

We simulate offline bandit learning (Algorithm 2), where feedback is collected all at once with the initial model. The learning scenario follows the previous section: only a limited amount of supervised data is available (64, 256, or 1,024 in-domain examples) to train initial models.

Table 1 shows the performance of offline simulation experiments compared to online simulations. We observe mixed results. On SQuAD, HotpotQA, NQ, and NewsQA, offline learning outperforms online learning when using stronger initial models (i.e., models trained on 256 and 1,024 examples). This illustrates the benefit of the more standard training loop, especially with our Transformer-based model that is better optimized with a linear learning rate schedule and multiple epochs, both incompatible with the online setup. On TriviaQA and SearchQA, offline simulation is ineffective regardless of the performance of initial models. This result echoes the learning challenges in the online counterparts on these two datasets.

Online vs. Offline Regret Table 2 compares online and offline regret. Regret numbers are averaged over the number of feedback observations.[11] Online learning generally displays lower regret for similar initial models on SQuAD, HotpotQA, and NQ. This is expected because later interactions in the simulation can benefit from early feedback in online learning. In contrast, in our offline scenario, we only update after seeing all examples, so regret numbers depend on the initial model only. Regret results on NewsQA, TriviaQA, and SearchQA are counterintuitive, generally showing that online learning has similar or higher regret. The cases showing significantly higher online regret (64+sim on NewsQA and SearchQA) can be explained by the learning failing, which impacts online regret, but not our offline regret. The others are more complex, and we hypothesize that they may be because of combination of (a) inherent noise in the data; and (b) in cases where online learning is effective, the gap between the strictly-defined reward that is used to compute regret and the relaxed F1 evaluation metric. Further analysis is required for a more conclusive conclusion.

7 Domain Adaptation

Learning from user feedback creates a compelling avenue to deploy systems that target new domains not addressed by existing datasets. The scenario we simulate in this section starts with training a QA model on a complete existing annotated dataset, and deploying it to interact with users and learn from their feedback in a new domain. We do not assume access to any annotated training data in

[11] Table 8 in Appendix B lists the percentage of positive feedback in online and offline in-domain simulation.

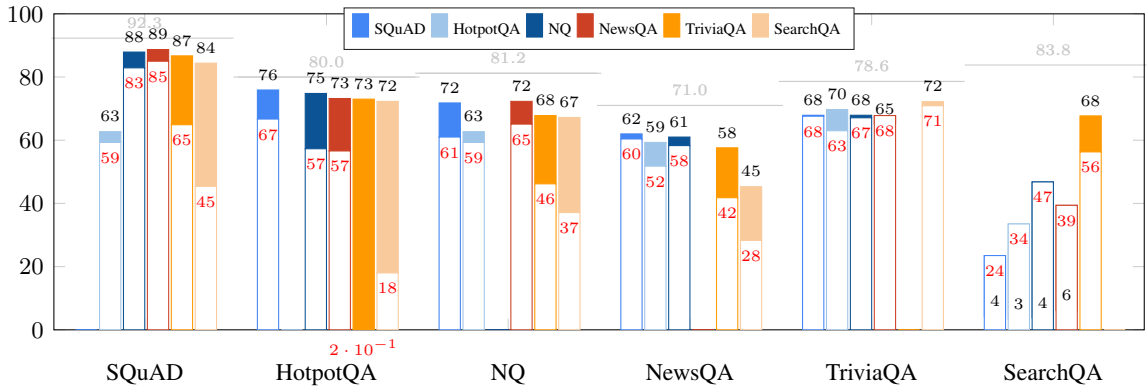

Figure 4: Online domain adaptation simulation development F1 performance. Horizontal grey lines represent the supervised training performance on each complete dataset. Bar colors denotes the source domain. Labels in red are the performance of initial models on the target domain (x-axis). Solid colors and black labels represent simulation performance on the target domain.

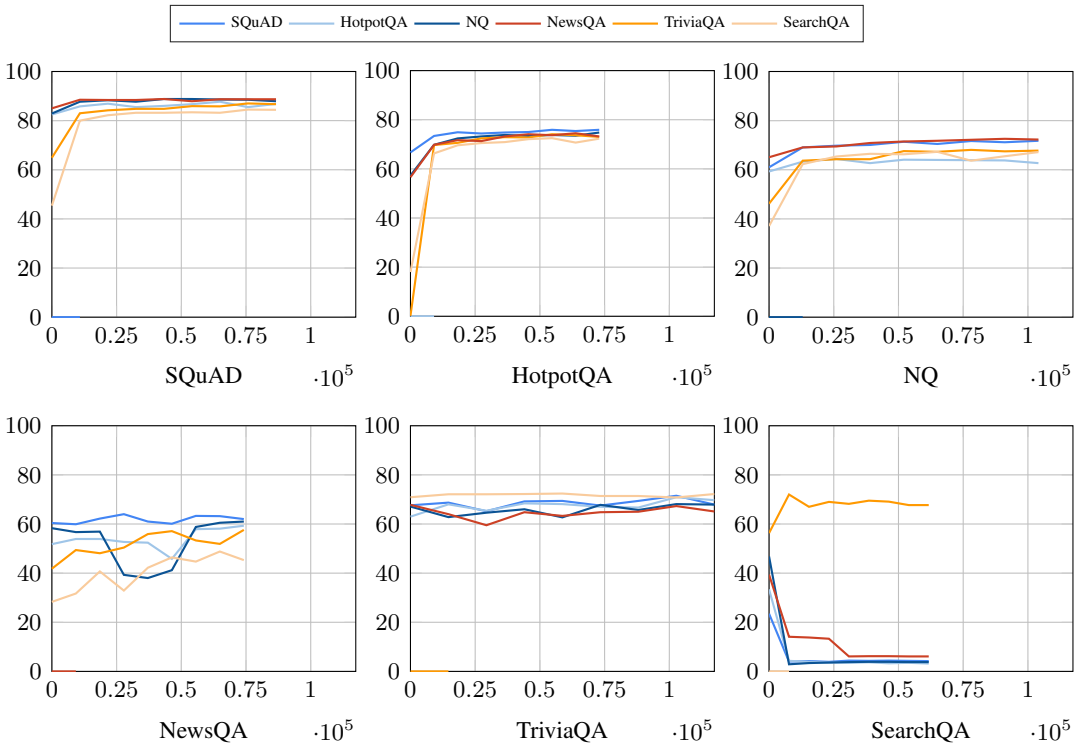

Figure 5: Online domain adaptation simulation development F1 learning curves. X-axis is the number of examples with feedback observed. Colors denote the source domain.

the target domain. We report experiments with online learning. Offline adaptation experiments are discussed in Appendix B.3.

Figure 4 shows online domain adaptation performance. On 22/30 configurations, online adaptation introduces significant performance gains (>2 F1 score). For example, adapting from TriviaQA and SearchQA to the other four domains improves performance by 27–72.8 F1. On HotpotQA, the model initially trained on TriviaQA shows an impressive adaptation, improving from 0.2 F1 to 73 F1.[12]

Our simulations show reduced effectiveness when the target domain is either TriviaQA or SearchQA, likely because the simulated feedback is based on noisy distantly supervised data. For SearchQA, the low performance of initial models from other domains may also contribute to the adaptation failure. As expected, this indicates the effectiveness of the process depends on the relation

[12]We replicate this result with different model initializations to confirm it is not random.

Dataset	In-domain	SQuAD-initialized
HotpotQA	66.2 → 70.5	**66.7** → **75.9**
NQ	**61.8** → 67.9	61.0 → **71.8**
NewsQA	55.1 → 56.3	**60.4** → **62.0**
TriviaQA	34.2 → 62.1	**67.6** → **67.9**
SearchQA	**65.0** → **70.3**	23.5 → 4.2

Table 3: Online learning development F1 Comparison between in-domain with initial models trained on 1,024 supervised examples, and adaptation with SQuAD as the source domain. Each entry provide performance before (right-side of arrow) and after (left-side) feedback simulation. Higher before/after performance is in bold.

between the source and target domains. SearchQA seems farthest from the other domains, mirroring observations from prior work (Su et al., 2019).

Figure 5 shows learning curves for our simulation experiments. Generally, we observe the choice of source and target domains influences adaptation rates. Models quickly adapt to SQuAD, HotpotQA, and NQ, reaching near final performance with a quarter of the total feedback provided. On NewsQA, models initially trained on TriviaQA and SearchQA adapt slower than those initially trained on other three datasets. On TriviaQA, we observe little change in performance throughout simulation. On SearchQA, only the model initially trained on TriviaQA shows a performance gain. Both SearchQA and TriviaQA include context paragraphs from the web, potentially making domain adaptation from one to the other easier.

Lastly, we compare bandit learning with initial models trained on a small amount of in-domain data (Section 5) and initial models trained on a large amount of out-of-domain data. Table 3 compares online learning with initial models trained on 1,024 in-domain supervised examples and online domain adaptation with a SQuAD-initialized model. SQuAD initialization provides a robust starting point for all datasets except SearchQA. On four out of five datasets, the final performance is better with SQuAD-initialized model. This is potentially because the model is exposed to different signals from two datasets and overall sees more data, either as supervised examples or through feedback. However, on SearchQA, learning with SQuAD-initialized model performs much worse than learning with the initial model trained on 1,024 in-domain examples, potentially because of the gap in initial model performance (23.5 vs. 65 F1).

8 Related Work

Bandit learning has been applied to a variety of NLP problems including neural machine translation (NMT; Sokolov et al., 2017; Kreutzer et al., 2018a,b; Mendoncca et al., 2021), structured prediction (Sokolov et al., 2016), semantic parsing (Lawrence and Riezler, 2018), intent recognition (Falke and Lehnen, 2021), and summarization (Gunasekara et al., 2021). Explicit human feedback has been studied as a direct learning signal for NMT (Kreutzer et al., 2018b; Mendoncca et al., 2021), semantic parsing (Artzi and Zettlemoyer, 2011; Lawrence and Riezler, 2018), and summarization (Stiennon et al., 2020). Nguyen et al. (2017) simulates bandit feedback to improve an MT system fully trained on a large annotated dataset, including analyzing robustness to feedback perturbations. Our work shows that simulated bandit feedback is an effective learning signal for extractive question answering tasks. Our work differs in focus on reducing annotation costs by relying on few annotated examples only to train the initial model, or by eliminating the need for in-domain annotation completely by relying on data in other domains to train initial models. Implicit human feedback, where feedback is derived from human behavior rather than explicitly requested, has also been studied, including for dialogue (Jaques et al., 2020) and instruction generation (Kojima et al., 2021). We focus on explicit feedback, but implicit signals also hold promise to improve QA systems.

Alternative forms of supervision for QA have been explored in prior work, such as explicitly providing fine-grained information (Dua et al., 2020; Khashabi et al., 2020a). Kratzwald et al. (2020) resembles our setting in seeking binary feedback to replace span annotation, but their goal is to create supervised data more economically. Campos et al. (2020) proposes feedback-weighted learning to improves conversational QA using simulated binary feedback. Their approach relies on multiple samples (i.e., feedback signals) per example, training for multiple epochs online by re-visiting the same questions repeatedly, and tuning two additional hyperparameters. In contrast, we study improving QA systems via feedback as a bandit learning problem. In both online and offline setups, we assume only one feedback sample per example. We also provide extensive sensitivity studies to the amount of annotations available, different model initialization, and noisy feedback across various datasets.

Domain adaptation for QA has been widely studied (Fisch et al., 2019; Khashabi et al., 2020b), including using data augmentation (Yue et al., 2021), adversarial training (Lee et al., 2019), contrastive method (Yue et al., 2021), back-training (Kulshreshtha et al., 2021), and exploiting small lottery subnetworks (Zhu et al., 2021).

9 Conclusion

We present a simulation study of learning from user feedback for extractive QA. We formulate the problem as contextual bandit learning. We conduct experiments to show the effectiveness of such feedback, the robustness to feedback noise, the impact of initial model performance, the trade-offs between online and offline learning, and the potential for domain adaptation. Our study design emphasizes the potential for reducing annotation costs by annotating few examples or by utilizing existing datasets for new domains.

We intentionally adopt a basic setup, including a simple binary reward and vanilla learning algorithms, to illustrate what can be achieved with a relatively simple variant of the contextual bandit learning scenario. Our results already indicate the strong potential of learning from feedback, which more advanced methods are likely to further improve. For example, the balance between online and offline learning can be further explored using proximal policy optimization (PPO; Schulman et al., 2017) or replay memory (Mnih et al., 2015). With well-designed interface, human users may be able to provide more sophisticated feedback (Lamm et al., 2021), which will provide a stronger signal compared to our binary reward.

Our aim in this study is to lay the foundation for future work, by formalizing the setup and showing its potential. This is a critical step in enabling future research, especially going beyond simulation to study using *real* human feedback for QA systems. Another important direction for future work is studying user feedback for QA systems that do both context retrieval and answer generation (Lewis et al., 2020), where assigning the feedback to the appropriate stage in the process poses a challenge. Beyond extractive QA, we hope our work will inspire research of user feedback as a signal to improve other types of NLP systems.

Legal and Ethical Considerations

Our work's limitations are discussed in Section 1 and Section 9. All six datasets we use are from prior work, are publicly available, and are commonly used for the study of extractive QA. Section 4 reports our computational budget and experimental setup in detail. Our codebase is available at `https://github.com/lil-lab/bandit-qa`.

Acknowledgements

This research was supported by ARO W911NF-21-1-0106, NSF under grants No. 1750499, the NSF AI Institute for the Foundations of Machine Learning (IFML), and a Google Faculty Research Award. Finally, we thank the action editor and the anonymous reviewers for detailed comments.

References

Yoav Artzi and Luke Zettlemoyer. 2011. Bootstrapping semantic parsers from conversations. In *EMNLP*.

Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. 2002. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77.

Alberto Bietti, Alekh Agarwal, and John Langford. 2018. A contextual bandit bake-off. *JMLR*.

Jon Ander Campos, Kyunghyun Cho, Arantxa Otegi, Aitor Soroa Etxabe, Gorka Azkune, and Eneko Agirre. 2020. Improving conversational question answering systems after deployment using feedback-weighted learning. *COLING 2020*.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. Quac: Question answering in context. *EMNLP*.

Dheeru Dua, Sameer Singh, and Matt Gardner. 2020. Benefits of intermediate annotations in reading comprehension. *ACL*.

Matthew Dunn, Levent Sagun, Mike Higgins, V. Ugur Güney, Volkan Cirik, and Kyunghyun Cho. 2017. Searchqa: A new q&a dataset augmented with context from a search engine. *CoRR*.

Tobias Falke and Patrick Lehnen. 2021. Feedback attribution for counterfactual bandit learning in multi-domain spoken language understanding. *EMNLP*.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. MRQA 2019 shared task: Evaluating generalization in reading comprehension. *MRQA@EMNLP*.

Chulaka Gunasekara, Guy Feigenblat, Benjamin Sznajder, Ranit Aharonov, and Sachindra Joshi. 2021. Using question answering rewards to improve abstractive summarization. *Findings@EMNLP*.

Daniel G. Horvitz and D. J. Thompson. 1952. A generalization of sampling without replacement from a finite universe. *Journal of the American Statistical Association*.

Natasha Jaques, Judy Hanwen Shen, Asma Ghandeharioun, Craig Ferguson, Agata Lapedriza, Noah Jones, Shixiang Gu, and Rosalind Picard. 2020. Human-centric dialog training via offline reinforcement learning. *EMNLP*.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *TACL*.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *ACL*.

Daniel Khashabi, Tushar Khot, and Ashish Sabharwal. 2020a. More bang for your buck: Natural perturbation for robust question answering. *EMNLP*.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter E. Clark, and Hannaneh Hajishirzi. 2020b. Unifiedqa: Crossing format boundaries with a single qa system. *Findings@EMNLP*.

Noriyuki Kojima, Alane Suhr, and Yoav Artzi. 2021. Continual learning for grounded instruction generation by observing human following behavior. *TACL*.

Bernhard Kratzwald, Stefan Feuerriegel, and Huan Sun. 2020. Learning a cost-effective annotation policy for question answering. *EMNLP*.

Julia Kreutzer, Shahram Khadivi, Evgeny Matusov, and Stefan Riezler. 2018a. Can neural machine translation be improved with user feedback? *NAACL*.

Julia Kreutzer, Joshua Uyheng, and Stefan Riezler. 2018b. Reliability and learnability of human bandit feedback for sequence-to-sequence reinforcement learning. *ACL*.

Devang Kulshreshtha, Robert Belfer, Iulian Serban, and Siva Reddy. 2021. Back-training excels self-training at unsupervised domain adaptation of question generation and passage retrieval. *EMNLP*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc V. Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *TACL*.

Matthew Lamm, Jennimaria Palomaki, Chris Alberti, Daniel Andor, Eunsol Choi, Livio Baldini Soares, and Michael Collins. 2021. Qed: A framework and dataset for explanations in question answering. *TACL*.

John Langford and Tong Zhang. 2007. The epoch-greedy algorithm for contextual multi-armed bandits. *NeurIPS*.

Carolin Lawrence and Stefan Riezler. 2018. Improving a neural semantic parser by counterfactual learning from human bandit feedback. *ACL*.

Carolin (Haas) Lawrence, Artem Sokolov, and Stefan Riezler. 2017. Counterfactual learning from bandit feedback under deterministic logging : A case study in statistical machine translation.

Angeliki Lazaridou, Adhiguna Kuncoro, Elena Gribovskaya, Devang Agrawal, Adam Liska, Tayfun Terzi, Mai Gimenez, Cyprien de Masson d'Autume, Tomás Kociský, Sebastian Ruder, Dani Yogatama, Kris Cao, Susannah Young, and Phil Blunsom. 2021. Mind the gap: Assessing temporal generalization in neural language models. *NeurIPS*.

Seanie Lee, Donggyu Kim, and Jangwon Park. 2019. Domain-agnostic question-answering with adversarial training. *MRQA@EMNLP*.

Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *NeurIPS*.

Vania Mendoncca, Ricardo Rei, Luísa Coheur, A. Sardinha, Ana L'ucia Santos INESC-ID Lisboa, Instituto Superior T'ecnico, AI Unbabel, Centro de Lingu'istica da Universidade de Lisboa, and Faculdade de Ciencias de Universidade de Lisboa. 2021. Online learning meets machine translation evaluation: Finding the best systems with the least human effort. *ACL/IJCNLP*.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, and Georg Ostrovski. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540).

Khanh Nguyen, Hal Daumé, and Jordan L. Boyd-Graber. 2017. Reinforcement learning for bandit neural machine translation with simulated human feedback. *EMNLP*.

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

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

Ori Ram, Yuval Kirstain, Jonathan Berant, Amir Globerson, and Omer Levy. 2021. Few-shot question answering by pretraining span selection. *ACL*.

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

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. *ICLR*.

Artem Sokolov, Julia Kreutzer, Christopher Lo, and Stefan Riezler. 2016. Learning structured predictors from bandit feedback for interactive nlp. *ACL*.

Artem Sokolov, Julia Kreutzer, Kellen Sunderland, Pavel Danchenko, Witold Szymaniak, Hagen Fürstenau, and Stefan Riezler. 2017. A shared task on bandit learning for machine translation. *WMT*.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan J. Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. 2020. Learning to summarize from human feedback. *NeurIPS*.

Dan Su, Yan Xu, Genta Indra Winata, Peng Xu, Hyeon-Jin Kim, Zihan Liu, and Pascale Fung. 2019. Generalizing question answering system with pre-trained language model fine-tuning. *EMNLP*.

Richard S. Sutton and Andrew G. Barto. 1998. *Reinforcement learning: An introduction*. MIT press.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. Newsqa: A machine comprehension dataset. *Rep4NLP@ACL*.

Ronald J. Williams. 2004. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2020. Huggingface's transformers: State-of-the-art natural language processing. *EMNLP*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *EMNLP*.

Ziyu Yao, Yiqi Tang, Wen-tau Yih, Huan Sun, and Yu Su. 2020. An imitation game for learning semantic parsers from user interaction. *EMNLP*.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. *ICLR*.

Zhenrui Yue, Bernhard Kratzwald, and Stefan Feuerriegel. 2021. Contrastive domain adaptation for question answering using limited text corpora. *EMNLP*.

Michael J.Q. Zhang and Eunsol Choi. 2021. SituatedQA: Incorporating extra-linguistic contexts into QA. *EMNLP*.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav. Artzi. 2021. Revisiting few-sample bert fine-tuning. *ICLR*.

Haichao Zhu, Zekun Wang, Heng Zhang, Ming Liu, Sendong Zhao, and Bing Qin. 2021. Less is more: Domain adaptation with lottery ticket for reading comprehension. *Findings@EMNLP*.

Dataset	arg max	Sampling
SQuAD	80.0	73.6
HotpotQA	65.7	56.8
NQ	64.8	62.9

Table 4: Comparison of final F1 development scores between arg max and sampling in online simulation with initial models trained on 256 supervised in-domain examples.

A Additional Discussion

Reward Function Intuitively, partial credit reward may improve learning over binary rewards. We experiment with using F1 score of the predicted answer span as a more refined feedback.[13] In practice, this does not introduce a stronger learning signal, potentially because the distribution over F1 scores is bimodal and focused on extreme values: around 85 % F1 scores are either 0 or 1 for predicted spans from a SQuAD-trained model on 8% NQ training data. We observe similar trends on all six datasets across all setups. Experiments with BLEU score (Papineni et al., 2002) as feedback show similar conclusion and distribution to F1 score.

Perturbation In practice, noise in feedback is likely to be more systematic than the statistical simplification which defines noise as the random percentage of wrong feedback. For example, prior work (Nguyen et al., 2017) on bandit neural machine translation (NMT) proposes that noisy human feedback is granular, high-variance, and skewed, which can be approximated by mathematical functions and shows to significantly impact the bandit NMT learning. We experiment with the three perturbation functions from Nguyen et al. (2017) on F1 reward. Our experiments show that the effect of adding these perturbation functions is negligible. We hypothesize that the reward distribution for NMT is likely to be closer to a normal distribution, rather than a bimodal one like QA.

B Additional Experiments

B.1 Method of Sampling

While arg max can bias towards exploitation, sampling can encourage more exploration. We experiment with prediction via arg max and sampling from the output distribution over spans. Table 4 shows that arg max performs better than random sampling on three datasets. This set of experiments is conducted with batch size 80.

B.2 Sensitivity Analysis

Table 5 shows the sensitivity analysis results for online in-domain simulation on HotpotQA and TriviaQA. We experiment with five initial models trained on different sets of 64 or 1,024 supervised examples, each used to initiate a separate simulation experiment. For weaker initial models trained on 64 supervised examples, four out of five experiments on HotpotQA show performance gains similar to our main results, except one experiment that starts with a very low initialization performance. Nearly all experiments on TriviaQA collapse (mean F1 of 7.3). Our sensitivity analysis with stronger initial models trained on 1,024 examples shows that the final performance is stable across runs on both HotpotQA and TriviaQA (standard deviations are 0.5 and 2.6).

B.3 Offline Adaptation

We perform domain adaptation with offline learning, and compare its performance with online adaptation. Table 6 shows the performance gain of offline adaptation simulation compared to the online setup. In most settings, online learning proves to be more effective, possibly because it observes feedback from partially adapted model predictions. In a few settings (4/30), we observe better adaptation with offline settings (+1.1 to +4.6). Overall, we observe that online learning is more effective on domain adaptation, while offline adaption performs slightly better when both domains are related (e.g., same source domain).

[13] We set the reward as -0.1 if receiving a 0 F1 score. In general, updating with negative rewards consistently shows a slightly higher performance across different setups for both binary and F1 reward.

Setup	64 + sim		1,024 + sim	
	HotpotQA	TriviaQA	HotpotQA	TriviaQA
42	16.4 → 66.8	16.6 → 3.3	66.1 → 71.5	55.1 → 58.9
43	15.9 → 69.7	24.0 → 3.4	65.3 → 71.6	63.0 → 65.0
44	18.1 → 68.8	23.3 → 2.4	66.4 → 71.3	58.0 → 65.1
45	6.7 → 1.4	22.8 → 9.9	65.1 → 71.9	60.8 → 64.2
46	24.8 → 67.5	16.2 → 17.4	66.2 → 70.5	34.2 → 62.1
μ_σ	$16.4_{6.5} \rightarrow 54.8_{29.9}$	$20.6_{3.8} \rightarrow 7.3_{6.4}$	$65.8_{0.6} \rightarrow 71.4_{0.5}$	$54.0_{11.4} \rightarrow 63.1_{2.6}$

Table 5: Sensitivity analysis: development F1 scores of online in-domain simulation on HotpotQA and TriviaQA with initial models trained on 64 or 1,024 examples. Each row corresponds to a different random seed and a different set of initial model training examples. $x \rightarrow y$ denotes that the performance changes from x to y after the model learns from feedback. Bottom row reports the mean and standard deviation across the five runs.

Sim+Eval**Pre-Train**	**SQuAD**	**HotpotQA**	**NQ**	**NewsQA**	**TriviaQA**	**SearchQA**
SQuAD		88.1(+1.3)	89.0(+1.1)	85.9(-2.8)	78.2(-8.5)	81.3(-3.1)
HotpotQA	75.1(-0.8)		73.7(-1.1)	69.6(-3.6)	56.6(-16.4)	68.1(-4.2)
NQ	69.1(-2.7)	67.3(+4.6)		64.7(-7.6)	42.2(-25.6)	52.6(-14.6)
NewsQA	59.3(-2.7)	48.4(-10.9)	48.5(-12.5)		0.1(-57.5)	45.6(0.3)
TriviaQA	62.5(-5.4)	66.6(-3.1)	9.5(-58.4)	3.2(-61.9)		70.2(-2.0)

Table 6: Offline domain adaptation simulation development F1 performance. Numbers in parenthesis show the performance gain (green) or decrease (red) of offline learning compared to online learning (Figure 4). We omit offline adaptation to SearchQA because of our previous observation that all online adaptations to SearchQA fail.

Dataset	**Train**	**Dev**	**Question (Q)**	**Context (C)**	**Q ⫫ C**
SQuAD	86,588	10,507	Crowdsourced	Wikipedia	✗
HotpotQA	72,928	5,904	Crowdsourced	Wikipedia	✗
NQ	104,071	12,836	Search logs	Wikipedia	✓
NewsQA	74,160	4,212	Crowdsourced	News articles	✓
TriviaQA♠	61,688	7,785	Trivia	Web snippets	✓
SearchQA♠	117,384	16,980	Jeopardy	Web snippets	✓

Table 7: Dataset statistics. ♠-marked datasets use distant supervision to match questions and contexts. $Q \perp\!\!\!\perp C$ is true if the question was written independently from the passage used for context.

Setup	**SQuAD**	**HotpotQA**	**NQ**	**NewsQA**	**TriviaQA**	**SearchQA**
64+sim	0.43/0.05	0.54/0.14	0.32/0.18	0.03/0.22	0.30/0.30	0.01/0.30
256+sim	0.49/0.32	0.67/0.48	0.36/0.25	0.23/0.23	0.31/0.34	0.34/0.37
1024+sim	0.56/0.50	0.75/0.70	0.41/0.39	0.34/0.36	0.35/0.42	0.38/0.41

Table 8: Percentage of positive examples in online/offline in-domain simulation in one pass on the training set.