Learning a Cost-Effective Annotation Policy for Question Answering

Bernhard Kratzwald[♢♠] **Stefan Feuerriegel**[♢] **Huan Sun**[♠]
[♢] Chair of Management Information Systems, ETH Zurich
[♠] Department of Computer Science and Engineering, The Ohio State University
{bkratzwald, sfeuerriegel}@ethz.ch sun.397@osu.edu

Abstract

State-of-the-art question answering (QA) relies upon large amounts of training data for which labeling is time consuming and thus expensive. For this reason, customizing QA systems is challenging. As a remedy, we propose a novel framework for annotating QA datasets that entails learning a cost-effective annotation policy and a semi-supervised annotation scheme. The latter reduces the human effort: it leverages the underlying QA system to suggest potential candidate annotations. Human annotators then simply provide binary feedback on these candidates. Our system is designed such that past annotations continuously improve the future performance and thus overall annotation cost. To the best of our knowledge, this is the first paper to address the problem of annotating questions with minimal annotation cost. We compare our framework against traditional manual annotations in an extensive set of experiments. We find that our approach can reduce up to 21.1% of the annotation cost.

1 Introduction

Question answering (QA) based on textual content has attracted a great deal of attention in recent years (Chen et al., 2017; Lee et al., 2018, 2019; Xie et al., 2020). In order for state-of-the-art QA models to succeed in real applications (e.g., customer service), there is often a need for large amounts of training data. However, manually annotating such data can be extremely costly. For example, in many realistic scenarios, there exists a list of questions from real users (e.g., search logs, FAQs, service-desk interactions). Yet, annotating such questions is highly expensive (Nguyen et al., 2016; He et al., 2018; Kwiatkowski et al., 2019): it requires the screening of a text corpus to find the relevant document(s) and subsequently screening the document(s) to identify the answering text span(s).

Motivated by the above scenarios, we study cost-effective annotation for question answering, whereby we aim to *accurately*[1] annotate a given set of user questions *with as little cost as possible*. Generally speaking, there has been extensive research on how to reduce effort in the process of data labeling (Haffari et al., 2009). For example, active learning for a variety of machine learning and NLP tasks (Siddhant and Lipton, 2018) aims to select a small, yet highly informative, subset of samples to be annotated. The selection of such samples is usually coupled with a particular model, and thus, the annotated samples may not necessarily help to improve a different model (Lowell et al., 2019). In contrast, we aim to annotate *all* given samples at low cost and in a manner that can subsequently be used to develop any advanced model. This is particularly relevant in the current era, where a dataset often outlives a particular model (Lowell et al., 2019). Moreover, there has also been some research into learning from distant supervision (Xie et al., 2020) or self-supervision (Sun et al., 2019). Despite being economical, such approaches often produce inaccurate or noisy annotations. In this work, we seek to reduce annotation costs without compromising the resulting dataset quality.

We propose a novel annotation framework which learns a cost-effective policy for choosing between different annotation schemes, namely the conventional manual annotation scheme (MAN) and a semi-supervised annotation scheme (SEM). Unlike the manual scheme, SEM does not require humans to screen a text corpus or document(s) in order to retrieve annotations. Instead, it leverages an initialized QA system, which can predict top-n candidate annotations for documents or answer spans and asks humans to provide binary feedback (e.g., cor-

[1] By "accurate," we mean that the resulting annotations will be of a similar quality to those from conventional manual annotation.

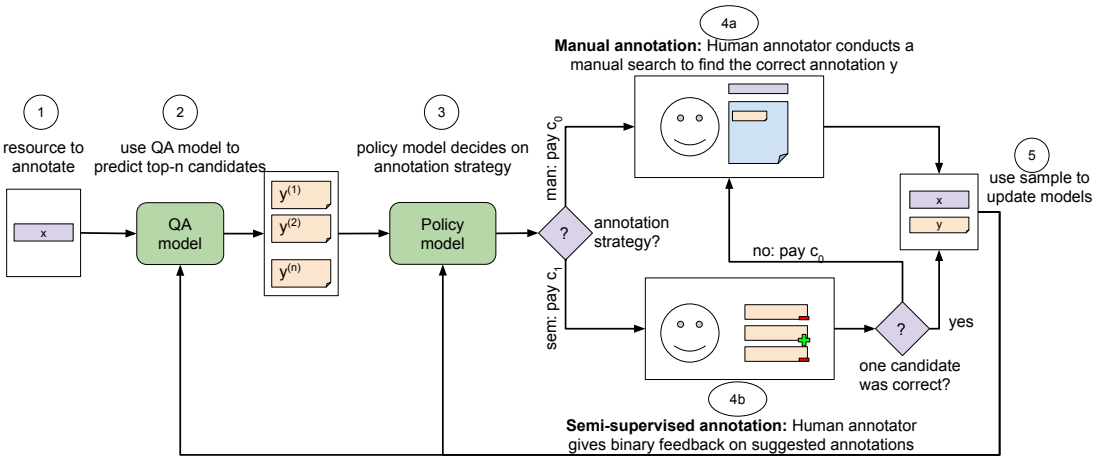

Figure 1: High-level overview of our framework: We leverage a QA model to predict candidate annotations for a given resource (e.g., x stands for a question or question-document pair, while y is the document or answer span). A policy model decides upon whether to invoke a MAN or SEM scheme based on those predictions. In the event that the semi-supervised strategy fails, we switch back to a manual annotation scheme. Finally, we use the annotated sample to update both the QA model and the policy model.

rect or incorrect) to the candidates. While this annotation scheme comes at a low cost, it fails when human annotators mark all candidates as incorrect. In such cases, the annotation cost has already been incurred and cannot be recouped. In order to produce an annotation, one must then draw upon the manual scheme (see Fig. 1), in which case the policy would have been more effective if it had chosen the manual annotation scheme instead. *Therefore, how to choose the best annotation scheme for each question is the challenge we must address for this task.*

To tackle the above challenge, we propose a novel approach for learning a cost-effective policy. Here the policy receives several candidates and decides on this basis which annotation scheme to invoke. We train the policy with a supervised objective and learn a cost-sensitive decision threshold. The inherent advantage of this method is that our policy immediately reacts to changing costs (without re-optimizing model parameters) and does not exceed the cost of conventional manual annotation. Our policy is updated iteratively as more annotations are obtained.

We compare our framework against conventional, manual annotations in an extensive set of experiments. We simulate the annotation of NaturalQuestions (Kwiatkowski et al., 2019), as it consists of real user questions from search logs. Models in our framework are initialized with an existing dataset (SQuAD, Rajpurkar et al., 2016) and, as more annotations on NaturalQuestions become available, the framework is continuously updated. We study the sensitivity of our framework to varying cost ratios between SEM and MAN. Our framework outperforms traditional manual annotation, even under conservative cost estimates for SEM, and in general reduces annotation costs in the range of 4.1% to 21.1%.

All source code is publicly available from github.com/bernhard2202/qa-annotation.

2 Related Work

Question answering: In this paper, we study cost-effective annotation for question answering over textual content. There have been extensive efforts to create large-scale datasets for text-based QA, which have facilitated the development of state-of-the-art neural network based models (e.g., Chen et al., 2017; Min et al., 2018; Lee et al., 2018; Kratzwald and Feuerriegel, 2018; Wang et al., 2018; Xie et al., 2020). Here we divide such datasets into two categories according to the way they were created: (1) Datasets whose questions were created by crowdsourcing during the annotation process. Prominent examples include the Stanford Question and Answer Dataset (SQuAD; Rajpurkar et al., 2016), HotPotQA (Yang et al., 2018), or NewsQA (Trischler et al., 2017). (2) "Natural" datasets in which real-world questions are a priori given. Here questions originate from, e.g., search logs or customer interactions. Prominent examples in this category include MS MARCO (Nguyen et al., 2016), DuReader (He et al., 2018), or Nat-

uralQuestions (Kwiatkowski et al., 2019). This paper focuses on the latter category, that is, annotating "natural" datasets in a more cost-effective fashion where a set of questions is given.

Active Learning: In the fields of machine learning and NLP, extensive research has been conducted on ways to reduce labeling effort (e.g., Zhu et al., 2008). For example, the objective of active learning is to select only a small subset that is highly informative (e.g., Haffari et al., 2009) for annotation. To this end, researchers have developed various techniques based on, e.g., model uncertainty (cf. Siddhant and Lipton, 2018), expected model change (Cai et al., 2013), or functions learned directly from data (e.g., Fang et al., 2017). However, the success of active learning is often coupled with a particular model and domain (Lowell et al., 2019). For instance, a dataset actively acquired with the help of an SVM model might underperform when used to develop an LSTM model. These problems become even more salient when complex black-box models are used in NLP tasks (cf. Chang et al., 2019). To summarize, active learning reduces annotation costs by deciding *which* samples should be annotated. In our approach, we aim to annotate *all* samples and study *how* we should annotate them in order to reduce costs. Thus, the two approaches are orthogonal and can be combined.

Learning from weak supervision and user feedback: Another approach to reducing annotation costs is changing full supervision to some form of weak (but potentially noisier) supervision. This has been adopted for various tasks such as machine translation (Saluja, 2012; Petrushkov et al., 2018; Clark et al., 2018; Kreutzer and Riezler, 2019), semantic parsing (Clarke et al., 2010; Liang et al., 2017; Talmor and Berant, 2018), or interactive systems that learn from user interactions (Iyer et al., 2017; Gur et al., 2018; Yao et al., 2019, 2020). For instance, Iyer et al. (2017) used users to flag incorrect SQL queries. In contrast, similar approaches for text-based question answering are scarce. Joshi et al. (2017) used noisy distant supervision to annotate the answer span and document for given trivia questions and their answers. Kratzwald and Feuerriegel (2019) designed a QA system that continuously learns from noisy user feedback after deployment. In contrast to these works, this paper studies the problem of reducing labeling cost while maintaining accurate annotations.

Quality estimation and answer triggering: In a broader sense, this work is related to the literature on translation quality estimation (e.g., Martins et al., 2017; Specia et al., 2013). The goal in such works is to estimate (and possibly improve) the quality of translated text. Similarly, in question answering researchers use means of quality estimation for answer triggering (Zhao et al., 2017; Kamath et al., 2020). Here, QA systems are given the additional option to abstain from answering a question when the best prediction is believed to be wrong. In our work, we estimate the quality of a set of suggested label candidates and, on the basis of these estimates we decide which annotation scheme to invoke.

3 Proposed Annotation Framework

We study the problem of reducing the overall cost for annotating every given question $[q_1, \ldots, q_m]$. Specifically, our objective is to obtain the corresponding question-document-answer triples $\langle q_i, d_i, s_i \rangle$. In this paper, the natural language question q_i is given, while we want to obtain the following annotations: the document from a text corpus $d_i \in \mathcal{D}$ that contains the answer and the correct answer span s_i within the document d_i.

3.1 Framework Overview

Fig. 1 provides an overview of our framework for a cost-effective annotation of QA datasets. The framework comprises two main components: a **QA model** is used to suggest candidates for a resource to annotate while a **policy model** decides which annotation scheme to invoke (i.e., action). Our framework makes use of two annotation schemes: a traditional **manual annotation scheme (MAN)** and our **semi-supervised annotation scheme (SEM)**. Both annotation schemes incur different costs and, hence, the learning task is to find and update a cost-effective policy π for making that decision.

QA model: We define Ω as an arbitrary QA model over a text corpus $\mathcal{D}$ with the following properties. First, the model can be trained from annotated data samples, e.g., $\Omega \leftarrow \text{train}\left(\{\langle q_i, d_i, s_i \rangle\}_{0<i<\ldots}\right)$. Second, for a given question the model can predict a number of top-n documents likely to contain the answer, i.e., $\Omega^D : q \rightarrow [d^{(1)}, \ldots, d^{(n)}] \in \mathcal{D}$. Third, for a given question-document pair the model can predict a number of top-n answer spans, i.e., $\Omega^S : \langle q, d \rangle \rightarrow [s^{(1)}, \ldots, s^{(n)}]$. These properties

are fulfilled by recent QA systems dealing with textual content (e.g., Chen et al., 2017; Wang et al., 2018).

Policy model: For every question, we distinguish two policy models: a policy model π^D responsible for annotating documents and π^S for answer spans. For brevity, we sometimes drop the superscripts S and D and simply refer to them as π. The policy models decide whether a manual annotation scheme or rather our proposed semi-supervised annotation scheme is used, each of which is associated to different costs.

3.2 Annotation Schemes

Manual annotation (MAN) scheme: This scheme represents the status quo in which all annotations are determined manually. In order to annotate a question q_i, a human annotator must first manually search through the text corpus $\mathcal{D}$ in order to identify the document d_i that answers the question. In a second step, a human annotator manually reads through the document d_i and marks the answer span s_i.

We assume separate costs, which are fixed over time, for every annotation-level. The price of annotating a document for a given question is defined as c_0^D and the price of annotating an answer span to a given question-document tuple as c_0^S. We explicitly distinguish these costs as the tasks can be of differing difficulty.

Semi-supervised annotation (SEM) scheme: This scheme is supposed to reduce human effort by presenting candidates for annotation, so that only simple binary feedback is needed. In particular, human annotators no longer need to search through the entire document or corpus. Instead, we use the QA model Ω to generate a set of candidates (e.g., top-ranked documents or answer spans) and ask human annotators to give binary feedback in response (e.g., accept the candidate or reject it). This replaces the complex search task with a simpler form of interaction. As an example, to annotate the answer span for a question-document pair, the human annotator would not be required to read the entire document d_i, but only to determine which of the top-n answers provided by $\Omega^S(\langle q_i, d_i\rangle)$ are correct. We assume SEM costs c_1^D to annotate a document and c_1^S to annotate an answer span.

The SEM scheme should make annotations more straightforward, as providing binary feedback requires less time than reading through the texts. Hence, we assume that $c_1^S < c_0^S$ and $c_1^D < c_0^D$ hold. However, semi-supervised annotations might fail when none of the candidates is correct (i.e., the human annotators reject all candidates). In this case, our framework must revert to the MAN procedure in order to obtain a valid annotation. As a consequence, the associated cost will increase to the accumulated cost for both the SEM and the MAN schemes.

Note that, no matter which scheme is chosen in practice, all annotations are confirmed by human annotators and our resulting dataset will be equal in quality to those resulting from traditional annotation.

3.3 Annotation Costs

Both annotation schemes, MAN and SEM, incur different costs that further vary depending on whether annotation is provided at document level (c^D) or at answer span level (c^S). For annotating documents, the cost amounts to

$$c^D(a|q_i, d_i^*) = \begin{cases} c_a^D, \text{ if } a = 0 \text{ or } (a = 1 \text{ and} \\ \qquad\qquad d_i^* \in \Omega^D(q_i)), \\ c_0^D + c_1^D, \quad \text{otherwise} \end{cases} \tag{1}$$

where $a = \{0, 1\}$ is the selected annotation scheme and d_i^* is the ground-truth document annotation. Hence, $d_i^* \in \Omega^D(q_i)$ indicates the candidate set contains the ground-truth annotation and SEM is successful.
For annotating answer spans, the cost is given by

$$c^S(a|\langle q_i, d_i\rangle, s_i^*) = \begin{cases} c_a^S, \text{ if } a = 1 \text{ or } (a = 0 \text{ and} \\ \qquad\qquad s_i^* \in \Omega^S(\langle q_i, d_i\rangle)), \\ c_0^S + c_1^S, \quad \text{otherwise.} \end{cases} \tag{2}$$

Alternatively, we can write the cost function as a matrix of annotation costs (Tbl. 1). The diagonal entries reflect the costs paid for choosing the optimal scheme. The off-diagonals refer to the costs paid for a sub-optimal method (misclassification costs).

4 Learning a Cost-Effective Policy

4.1 Objective

We aim to minimize overall annotation cost via

$$\sum_i \mathbb{E}_{\pi^D, \pi^S} \left[c^D(a|q_i, d_i^*) + c^S(a|\langle q_i, d_i\rangle, s_i^*) \right]. \tag{3}$$

Annotation Costs for a Document d		
	Cost-optimal: MAN	Cost-optimal: SEM
Selected: MAN	c_0^D	c_0^D
Selected: SEM	$c_0^D + c_1^D$	c_1^D

Annotation Costs for an Answer Span s		
	Cost-optimal: MAN	Cost-optimal: SEM
Selected: MAN	c_0^S	c_0^S
Selected: SEM	$c_0^S + c_1^S$	c_1^S

Table 1: Costs for annotating documents (top) and answer spans (bottom). The costs depend on the selected annotation scheme (rows) and the scheme that would have been cost-optimal (columns).

It is important to see that the QA model and the policy model are intertwined, with both having an impact on Eq. 3. Updating the policy models learns the trade-off between SEM and MAN annotations and, hence, directly minimizes the overall costs. Updating the QA model Ω increases the number of times suggested candidates are correct and, therefore, the fraction of successful SEM annotations. For instance, when adapting to a new domain, only a small fraction of suggested candidate annotations are correct, limiting the effectiveness of the SEM annotation. However, as we annotate more samples, we improve Ω and thus more suggested candidate annotations will be correct. For this, we later specify suitable updates for both the QA model and the policy model.

4.2 Annotation Procedure and Learning

Our framework proceeds according to these nine steps when annotating a question q_i (see Alg. 1): First, we predict a number of top-n documents that would be shown to annotators in the case of SEM annotation (line 2). Next, we decide upon the annotation scheme conditional on the prediction from the QA model (line 3) and, based on the selected scheme, request the ground-truth document annotation (line 4). After receiving the ground-truth document and observing the annotation costs, we update our policy network in line 5 (see Sec. 4.3). Next, we predict a number of top-n answer span candidates for the question-document pair (line 6) and then decide upon the annotation scheme in line 7. After receiving the answer span annotation and observing a cost (line 8), we again update our policy model (line 9). Finally, we update the QA model with the newly annotated training sample in line 10 (see Sec. 4.4). In practice, both policy updates and QA model updates (lines 5, 9, and 10) are invoked after a batch of questions is annotated. Furthermore, we initialize all models with an existing dataset (e.g., SQuAD).

Algorithm 1: High-Level Procedure of Annotation and Learning

Input : list of questions $[q_1, \ldots, q_m]$ text corpus $\mathcal{D}$; QA model Ω; policy models π^D and π^S
Result: annotated dataset $\{\langle q_i, d_i, s_i\rangle\}_{0<i<m}$

1 **while** $i \leq m$ **do**
2 $\quad [d^{(1)}, \ldots, d^{(n)}] \leftarrow \Omega^D(q_i)$; predict top-$n$ documents
3 $\quad a \leftarrow \pi^D(q_i, [d^{(1)}, \ldots, d^{(n)}])$; decide upon annotation scheme
4 $\quad d_i \leftarrow annotate(q_i|a)$; annotate document
5 $\quad$ update π^D w.r.t. the observed costs $c^D(a|q_i, d_i)$;
6 $\quad [s^{(1)}, \ldots, s^{(n)}] \leftarrow \Omega^S(\langle q_i, d_i\rangle)$; predict top-$n$ answer candidates
7 $\quad a \leftarrow \pi^S(q_i, [s^{(1)}, \ldots, s^{(n)}])$; decide upon annotation scheme
8 $\quad s_i \leftarrow annotate(\langle q_i, d_i\rangle|a)$; annotate answer span
9 $\quad$ update π^S w.r.t. the observed costs $c^S(a|\langle q_i, d_i\rangle, s_i)$;
10 $\quad$ update QA model Ω with $\langle q_i, d_i, s_i\rangle$
11 **end**

4.3 Policy Updates

Updating our policy model proceeds in three steps. (1) We calculate whether the chosen action for past annotations was cost-optimal (i.e., whether the policy should have chosen the other scheme for annotation or not). (2) We use this information to update the policy model with a supervised binary classification objective. This trains the policy to predict the probability of an annotation scheme given a new sample $p(a|x)$ without taking costs into account. (3) We find a cost-sensitive decision threshold that chooses the optimal action with respect to the costs. All three steps are repeated after a full batch of samples has been annotated.

Separating the policy update and the cost-sensitive decision threshold has several benefits. First, we know from cost-sensitive classification that we can calculate an optimal threshold point for ground-truth probabilities $p(a|x)$ (c.f. Elkan, 2001; Ting, 2000). Therefore, we can focus our effort on determining probabilities as accurate as possible. Second, the decision threshold is calculated only from the costs c_a^S and c_a^D and, hence, if costs change, we do not need to re-estimate parameters but can directly adjust our policy.

(1) Finding the cost-optimal action: In order to train the policy with a supervised update, we require labels for the cost-optimal annotation scheme

for a given sample. If we choose the SEM annotation scheme, we immediately know whether the action was cost-optimal or not. This is due to the fact that, if the semi-supervised annotation fails, we have to switch to the MAN scheme to receive an annotation and pay both costs. On the other hand, if we choose the MAN scheme, we can observe the optimal action only after receiving the ground-truth annotation: We can then simply run the QA model and validate whether the annotation was contained within the top-n candidates. If so, the SEM action would have been the better choice; otherwise, choosing MAN would have been cost-optimal.

(2) Supervised model updates: Our policy model is a neural network with parameters θ that predicts an annotation scheme for a given sample x, i.e.,

$$p(a|x) = \text{NN}_\theta(x). \tag{4}$$

Note that we dropped the S and D indices here as both policies differ only in the neural network architecture used. We can then simply train the policy with a supervised binary cross-entropy loss given the cost-optimal action that we calculated beforehand. Since the SEM scheme is often sub-optimal in the beginning, the training data is highly imbalanced. Therefore, we down-sample past annotations with a sampling ratio of α such that our training data is equally balanced.

(3) Cost-sensitive decision threshold: Choosing the annotation scheme with the highest probability does not take the actual costs into account. For instance, if SEM annotations are much cheaper than MAN annotations, we want to choose the semi-supervised scheme even if its probability for success is low. More formally, we want to choose the annotation scheme a that has the lowest expected cost $R(a|x)$, i.e.,

$$R(a \,|\, x) = \sum_{a'} p(a' \,|\, x)\, c(a, a') \tag{5}$$

where $c(a, a')$ is the annotation cost for choosing scheme a when the optimal scheme was a' (see Tab. 1). Since we used down-sampling in our training, we have to calibrate the probabilities $p(a|x)$ with the sampling ratio α (Pozzolo et al., 2015). Assuming the calibrated probabilities are accurate, there exists an optimal classification threshold β (Elkan, 2001) that minimizes Eq. 5 and Eq. 3. Therefore, we define our policy as follows:

$$\pi(a|x, \alpha, \beta) = \begin{cases} 1, \text{ if } \frac{\alpha\, p(a=1|x)}{(\alpha-1)\, p(a=1|x)+1} \geq \beta \\ 0, \text{ otherwise.} \end{cases} \tag{6}$$

The optimal β can be derived from the classification cost matrix (Elkan, 2001) via

$$\beta = \frac{c(1,0) - c(0,0)}{c(1,0) - c(0,0) + c(0,1) - c(1,1)}, \tag{7}$$

again omitting superscripts D and S for brevity. Eq. 7 is then simplified as the fraction of SEM annotation costs to MAN annotation costs. Therefore, we can derive β at document level

$$\beta^D = c_1^D / c_0^D, \tag{8}$$

and answer span level

$$\beta^S = c_1^S / c_0^S. \tag{9}$$

4.4 Model Updates

Periodically updating the QA model Ω allows the framework to adapt to the question style and domain at hand during the annotation process. Therefore, we improve the top-n accuracy and the success rate of the SEM scheme over time. For practical reasons, we refrain from updating Ω after every annotation but periodically retrain the model after a batch of samples is annotated. In order to update the QA model, we can use the fully annotated QA samples in combination with a supervised objective.

5 Experimental Setup

In this section, we introduce our experimental setup and implementation details.

5.1 Datasets

We base our experiments on the NaturalQuestion dataset (Kwiatkowski et al., 2019). We choose this dataset as it is composed of about 300,000 real user questions posed to the Google search engine along with human-annotated documents and answer spans. Simulating the annotation of this dataset is similar to what would happen for domain customization of QA models in real practice (e.g., for search logs, FAQs, logs of past customer interactions). We focus on questions from the training-split that possess an answer span annotation and leave the handling of questions that do not have an answer for future work. The corpus for annotations

is fixed to the English Wikipedia,[2] containing more than 5 million text documents.

Simulation of annotations: Annotations in our experiments are simulated from the original dataset. If the framework chooses MAN annotation, we simply use the original annotation from the dataset. If a SEM annotation is chosen, we simulate users that give positive feedback only to the ground-truth document and to the answer spans where the text matches[3] the ground-truth annotation. We then construct the new annotation using the candidate with positive feedback. Since we simulate annotations, we conduct extensive experiments on how annotation costs influence the performance of our framework.

5.2 Baselines

To the best of our knowledge, there is no comparable prior work. Owing to this fact, we evaluate our framework against several customized baselines. First, we compare our approach against a manual annotation baseline in which we always invoke the full MAN method to annotate samples. This represents the traditional method of annotating QA datasets and thus our prime baseline. Second, we draw upon a clairvoyant oracle policy that always knows the optimal annotation method. We use this baseline to report an upper bound of the savings that our framework could theoretically achieve. Third, we use our framework without updates on the QA model Ω. This quantifies the cost-savings achieved by the interactive domain customization during annotation. Finally, we present a randomized baseline where the annotation scheme is decided by a randomized coin-toss.

5.3 Implementation Details

The QA model Ω is built as follows. We use a state-of-the-art BERT-based (Devlin et al., 2018) implementation of RankQA (Kratzwald et al., 2019). This combines a simple tf-idf-based information retrieval module with BERT as a module for machine comprehension. Both policy models π^D and π^S are implemented as three-layer feed-forward networks with dropout, ReLu activation, and a single output unit with sigmoid activation in the last layer. For the policy π^D, we use the information retrieval scores as input. For π^S, we use the statistical features of answer-span candidates as calculated by RankQA (Kratzwald et al., 2019) as input. We also experimented with convolutional neural networks directly on top of the last layer of BERT, but without yielding improvements that justified the additional model complexity. We initialize all models with the SQuAD dataset (see our supplements).

Hyperparameter setting: We set the number of candidates that are shown to annotators during a SEM annotation to $n = 5$. The policy networks decide upon the annotation method based on features of the $2n$ highest-ranked candidates, i.e., the top-10. The batch size for updates in Alg. 1 is set to 1,000 annotated questions. Details on hyperparameters of our QA and policy models are provided in the supplements.

6 Experimental Results

We group our experiments into three parts. First, we focus only on annotating the answer span for given question-document pairs, as this is the more challenging task.[4] Second, we carry out a sensitivity analysis in order to demonstrate how our framework adapts to different costs of SEM annotations and to show that we never exceed the cost of traditional annotation. Third, we evaluate our framework based on the annotation of a full dataset, including both answer span and document annotations, in order to quantify savings in practice by using our framework.

6.1 Performance on Answer Span Annotations

The annotation framework was used to annotate 45 batches of question-document pairs with the corresponding answer spans. The annotation costs are set to one price-unit for each MAN annotation and one third of the unit for each SEM annotation. (In the next section, we carry out an extensive sensitivity analysis where the ratio for annotation costs between MAN and SEM is varied.)

In Fig. 2 (left), we plot the average annotation costs in every batch with a dashed line, together with a running mean depicted as a solid line. Compared to conventional, manual annotation, our framework successfully reduces annotation cost by around 15% after only 20 batches. We further

[2] We extracted articles from the Wikipedia dump collected in October 2019, as this is close to the time period in which the NaturalQuestions dataset was constructed.

[3] Here we only count exact matches.

[4] Manually finding an answer span involves reading a document in depth. Manual document annotation is easier, as it can be supported with tools such as search engines. In such a case, our framework could still be used for answer span annotation, as is shown in Section 6.1.

compare it with an oracle policy that always picks the best annotation method. The latter provides a hypothetical upper bound according to which approximately 40–45% of annotation cost could be saved. Finally, we show the performance of our framework without updates of the QA model Ω. Here we can see that its improvement over time is lower, as the framework is not capable of adapting to the question style and domain used during annotation. In sum, our framework is highly effective in reducing annotation cost.

Fig. 2 (right) shows how many samples we could annotate (y-axis) with a restricted budget (x-axis). For instance, assume we have a budget of 40k price units available for annotation. Conventional, manual annotation would result in exactly 40k annotated samples as we fixed the cost for each MAN annotation to one unit. With the same budget, our annotation framework with semi-supervised annotations succeeds in annotating an additional ∼9,000 samples.

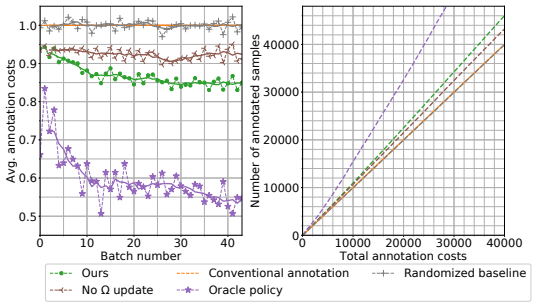

Figure 2: Left: average annotation costs in every batch as a dashed line, together with a running mean as a solid line. Right: how many samples we could annotate (y-axis) with a restricted available budget (x-axis).

6.2 Cost-Performance Sensitivity Analysis

The advantage of our framework over manual annotations depends on the cost ratio between the SEM and MAN schemes. In order to determine this, we identify the cost-range in which our framework is profitable as a function of SEM annotation costs. We study this via the following experiment: we repeatedly annotate 40k samples and keep the MAN annotation costs fixed to one price unit, while we increase the costs of smart annotations from 0.05 to 0.95 in increments of 0.05. Finally, we measure the average annotation costs for a single sample; see Fig. 3 (left).

Fig. 3 (left) demonstrates that our framework effectively lowers annotation costs when the price for SEM annotations drops below 0.6 as compared to manual annotations, which are fixed to one price-unit. Most notably, even when SEM annotations become expensive and almost equal the costs of MAN annotations, the average annotation costs do not exceed those of strictly manual annotation. This can be attributed to our cost-sensitive decision threshold, which does not require exploration as in reinforcement learning, but directly sets the threshold in Eq. 6 sufficiently high.

In Fig. 3 (right), we again show the number of samples that were annotated with a restricted budget of 40k price units. We marked the absolute gain in number of samples over traditional annotation in the plot. The benefit of our framework becomes evident once again when the ratio of SEM annotation costs to MAN annotations costs falls below 0.6.

To summarize, our framework is highly cost-effective: it reduces overall annotation costs or, alternatively, increases the number of annotated samples under a restricted budget if annotation costs of SEM are approximately half those of MAN. If the costs are less than half those of MAN annotation, the benefits are especially pronounced. Even if this assumption does not hold, our framework never exceeds the costs of manual annotation and never results in fewer annotated samples.

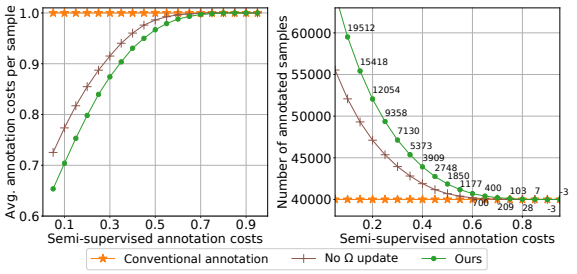

Figure 3: Left: average annotation costs when varying the ratio of SEM over MAN annotation costs. Right: number of samples that were annotated with a restricted budget for a given SEM annotation cost.

6.3 Performance on Full Dataset Annotation

In the last experiment, we simulate a complete annotation of the NaturalQuestions dataset, including annotations at both document level and answer span level. By annotating a complete dataset, we want to quantify the savings of our framework in practice. We again set the cost of each MAN annotation to one price-unit and repeated the experiment three times by setting the SEM annotation cost (c_1)

	Document-level			Answer span-level			Overall		
	$c_1 = 1/4$	$c_1 = 1/3$	$c_1 = 1/2$	$c_1 = 1/4$	$c_1 = 1/3$	$c_1 = 1/2$	$c_1 = 1/4$	$c_1 = 1/3$	$c_1 = 1/2$
Traditional Annotation	102.4	102.4	102.4	102.4	102.4	102.4	204.8	204.8	204.8
Ours	79.8	85.6	98.0	81.8	87.0	98.3	161.6	172.7	196.3
	(22.1%)	**(14.9%)**	**(4.2%)**	**(20.1%)**	**(14.9%)**	**(4.0%)**	**(21.1%)**	**(15.7%)**	**(4.1%)**

Table 2: Overall cost ($\times 10^3$ price unit) for annotating the NaturalQuestions dataset using our framework vs. conventional manual annotation for different SEM costs (c_1). Improvements are shown in parenthesis.

to one quarter, one third, and one half of the price unit. The results are shown in Tbl. 2. Depending on relative cost ratio c_1, we are able to save between 4.1% and 21.1% percent of the overall annotation cost. This amounts to a total of 40,000 to 8,000 price units.[5]

7 Discussion and Future Work

We assume for the purposes of this study that questions have an answer span contained in a single document and leave an extension to multi-hop questions and unanswerable questions to future research. The robustness of our framework is demonstrated in an extensive set of simulations and experiments. We deliberately choose to leave experiments including real human annotators to future research for the following reason. Outcomes of such an experiment would be sensitive to the design of the user interface as well as the study design itself. In this paper, we want to put the emphasis on the methodological innovation of our framework and the novel annotation scheme itself.

On the other hand, experiments involving real users would provide valuable insights concerning the annotation costs and the quality of a dataset annotated with our method. Furthermore, it would be worth investigating how inter-annotator agreement or potential human biases manifest in traditional datasets as compared to those generated with our framework.

8 Conclusion

We presented a novel annotation framework for question answering based on textual content, which learns a cost-effective policy to combine a manual annotation scheme with a semi-supervised annotation scheme. Our framework annotates all given questions accurately while limiting costs as much as possible. We show that our framework never incurs higher costs than traditional manual annotation. On the contrary, it achieves substantial savings. For example, it reduces the overall costs by about 4.1% when SEM annotations cost about half of MAN annotations. When that ratio is lowered to one fourth, our framework can reduce the total costs by up top 21.1%. We think that our framework could contribute to more accessible annotation of datasets in the future and possibly even be extended to other fields and applications in natural language processing.

[5] In these experiments, we obtain the overall cost by directly adding the costs of the two levels. Note that the cost can be different for document and answer span annotations, and that in such cases, our framework can still save costs at each level as shown in the table, although we cannot directly add up the costs as an overall sum.

Acknowledgments

We would like to thank the anonymous reviewers for their helpful comments. We are also deeply grateful to Ziyu Yao for her valuable feedback and help in brainstorming and formalizing this idea. This research was partly supported by the Army Research Office under cooperative agreements W911NF-17-1-0412, and by NSF Grant IIS1815674 and NSF CAREER #1942980. The views and conclusions contained herein are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notice herein.

References

W. Cai, Y. Zhang, and J. Zhou. 2013. Maximizing expected model change for active learning in regression. In *IEEE International Conference on Data Mining*, pages 51–60.

Haw-Shiuan Chang, Shankar Vembu, Sunil Mohan, Rheeya Uppaal, and Andrew McCallum. 2019. Overcoming practical issues of deep active learning and its applications on named entity recognition. *arXiv preprint arXiv:1911.07335*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Association for Computational Linguistics (ACL)*.

Kevin Clark, Minh-Thang Luong, Christopher D. Manning, and Quoc Le. 2018. Semi-supervised sequence modeling with cross-view training. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1914–1925.

James Clarke, Dan Goldwasser, Ming-Wei Chang, and Dan Roth. 2010. Driving semantic parsing from the world's response. In *Conference on Computational Natural Language Learning (CoNLL)*, pages 18–27.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics (NACL)*.

Charles Elkan. 2001. The foundations of cost-sensitive learning. In *International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'01, page 973–978.

Meng Fang, Yuan Li, and Trevor Cohn. 2017. Learning how to active learn: A deep reinforcement learning approach. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 595–605.

Izzeddin Gur, Semih Yavuz, Yu Su, and Xifeng Yan. 2018. DialSQL: Dialogue based structured query generation. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1339–1349.

Gholamreza Haffari, Maxim Roy, and Anoop Sarkar. 2009. Active learning for statistical phrase-based machine translation. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics (NACL)*, pages 415–423.

Wei He, Kai Liu, Jing Liu, Yajuan Lyu, Shiqi Zhao, Xinyan Xiao, Yuan Liu, Yizhong Wang, Hua Wu, Qiaoqiao She, Xuan Liu, Tian Wu, and Haifeng Wang. 2018. DuReader: a Chinese machine reading comprehension dataset from real-world applications. In *Workshop on Machine Reading for Question Answering*, pages 37–46.

Srinivasan Iyer, Ioannis Konstas, Alvin Cheung, Jayant Krishnamurthy, and Luke Zettlemoyer. 2017. Learning a neural semantic parser from user feedback. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 963–973.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1601–1611.

Amita Kamath, Robin Jia, and Percy Liang. 2020. Selective question answering under domain shift. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 5684–5696.

Bernhard Kratzwald, Anna Eigenmann, and Stefan Feuerriegel. 2019. Rankqa: Neural question answering with answer re-ranking. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Bernhard Kratzwald and Stefan Feuerriegel. 2018. Adaptive document retrieval for deep question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 576–581.

Bernhard Kratzwald and Stefan Feuerriegel. 2019. Learning from On-Line User Feedback in Neural Question Answering on the Web. In *The World Wide Web Conference (WWW)*, pages 906–916.

Julia Kreutzer and Stefan Riezler. 2019. Self-regulated interactive sequence-to-sequence learning. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 303–315.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics (TACL)*.

Jinhyuk Lee, Seongjun Yun, Hyunjae Kim, Miyoung Ko, and Jaewoo Kang. 2018. Ranking Paragraphs for Improving Answer Recall in Open-Domain Question Answering. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 565–569.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 6086–6096.

Chen Liang, Jonathan Berant, Quoc Le, Kenneth D. Forbus, and Ni Lao. 2017. Neural symbolic machines: Learning semantic parsers on Freebase with weak supervision. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 23–33.

David Lowell, Zachary C. Lipton, and Byron C. Wallace. 2019. Practical obstacles to deploying active learning. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 21–30.

André F. T. Martins, Marcin Junczys-Dowmunt, Fabio N. Kepler, Ramón Astudillo, Chris Hokamp, and Roman Grundkiewicz. 2017. Pushing the limits of translation quality estimation. *Transactions of the*

Association for Computational Linguistics (TACL), 5:205–218.

Sewon Min, Victor Zhong, Richard Socher, and Caiming Xiong. 2018. Efficient and Robust Question Answering from Minimal Context over Documents. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1725–1735.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Pavel Petrushkov, Shahram Khadivi, and Evgeny Matusov. 2018. Learning from chunk-based feedback in neural machine translation. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 326–331.

A. D. Pozzolo, O. Caelen, R. A. Johnson, and G. Bontempi. 2015. Calibrating probability with undersampling for unbalanced classification. In *2015 IEEE Symposium Series on Computational Intelligence*, pages 159–166.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *Empirical Methods in Natural Language Processing (EMNLP) Language Processing*, pages 2383–2392.

Avneesh Saluja. 2012. Machine Translation with Binary Feedback : a Large-Margin Approach. In *Biennial Conference of the Association for Machine Translation in the Americas*.

Aditya Siddhant and Zachary C. Lipton. 2018. Deep Bayesian Active Learning for Natural Language Processing: Results of a Large-Scale Empirical Study. *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2904–2909.

Lucia Specia, Kashif Shah, Jose G.C. de Souza, and Trevor Cohn. 2013. QuEst - a translation quality estimation framework. In *Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 79–84.

Yibo Sun, Duyu Tang, Nan Duan, Yeyun Gong, Xiaocheng Feng, Bing Qin, and Daxin Jiang. 2019. Neural semantic parsing in low-resource settings with back-translation and meta-learning. In *Association for the Advancement of Artificial Intelligence (AAAI)*.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *Conference of the North American Chapter of the Association for Computational Linguistic (NACL)*, pages 641–651.

Kai Ming Ting. 2000. A comparative study of cost-sensitive boosting algorithms. In *International Conference on Machine Learning (ICML)*, pages 983–990.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Workshop on Representation Learning for NLP*, pages 191–200.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerald Tesauro, Bowen Zhou, and Jing Jiang. 2018. R^3: Reinforced Reader-Ranker for Open-Domain Question Answering. In *Association for the Advancement of Artificial Intelligence (AAAI)*.

Yuqing Xie, Wei Yang, Luchen Tan, Kun Xiong, Nicholas Jing Yuan, Baoxing Huai, Ming Li, and Jimmy Lin. 2020. Distant supervision for multi-stage fine-tuning in retrieval-based question answering. In *The Web Conference (WWW)*, page 2934–2940.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 2369–2380.

Ziyu Yao, Yu Su, Huan Sun, and Wen-tau Yih. 2019. Model-based interactive semantic parsing: A unified framework and a text-to-SQL case study. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5447–5458.

Ziyu Yao, Yiqi Tang, Wen-tau Yih, Huan Sun, and Yu Su. 2020. An imitation game for learning semantic parsers from user interaction. *arXiv preprint arXiv:2005.00689*.

Jie Zhao, Yu Su, Ziyu Guan, and Huan Sun. 2017. An end-to-end deep framework for answer triggering with a novel group-level objective. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1276–1282.

Jingbo Zhu, Huizhen Wang, Tianshun Yao, and Benjamin K Tsou. 2008. Active learning with sampling by uncertainty and density for word sense disambiguation and text classification. In *International Conference on Computational Linguistics (Coling)*, pages 1137–1144.

Appendix

A Source Code

All source code is available from github.com/bernhard2202/qa-annotation.

B Details on the QA Model

We use the same hyperparemter configuration as reported in Kratzwald et al. (2019) without further fine-tuning. The model was initialized by training on the training split of the SQuADv1.1. dataset (Rajpurkar et al., 2016).

Parameter	Values
Dropout z	0.0, **0.3**, 0.5
Hidden units k	32 64 **128**
Learning rate	**0.0001**, 0.0005, 0.001
Epochs	15, 20, **25**

Table 3: Values used for gridsearch in hyperparameter tuning for the policy π^S

Parameter	Values
Dropout z	0.0, **0.3**, 0.5
Hidden units k	32 64 **128**
Learning rate	**0.0001**, 0.0005, 0.001
Epochs	15, 20, **25**

Table 4: Values used for gridsearch in hyperparameter tuning for the policy π^D

C Details on the Policy Model

The policy models π^D and π^S are implemented as feed forward networks composed of a dense layer with k output units and relu actvation, a dropout layer with dropout probability z, a second dense layer with $k/2$ outputs and relu activation, a dropout layer with dropout probability z, and a dense layer with a single output and sigmoid activation.

Initialization: for the first batch of annotations we initialize the policy models on SQuAD. After the first batch is annotated we only use the new data for policy updates.

Hyperparameter search: We tune hyperparamters on the SQuAD dataset using gridsearch with the values displayed in Tab. 3 and Tab. 4. Bold values mark final choices. We annotated the first 10 batches of SQuAD and choose the hyperparamters that had the lowest anntation cost. No hyperparemter tuning or architecture search was performed on the NaturalQuestions dataset which our experiments are based on.

D Estimation of Real Annotation Costs

In order to provide additional insights on the actual annotation costs involving real users we conducted a pre-test on Amazon MTURK. For this we showed a textual explanation of the MAN and SEM annotation scheme to workers and provided them with mockups for both inputs (answer-span annotation). Next, we asked 40 workers to report how much money they think would be a fair compensation for each of the tasks on a scale of one to ten. Workers reported on average a compensation of \$5.9 for MAN annotations and \$3.2 for SEM annotations. This ratio falls into the range where we make profits using our framework.

E System

All experiments were conducted with a Nvidia Titan Xp GPU on a Server with 192GB DDR4 RAM and two 10 Core Intel Xeon Silver 4210 2.2GHz Processors.