Crowdsourcing Human Annotation on Web Page Structure: Infrastructure Design and Behavior-Based Quality Control

SHUGUANG HAN, University of Pittsburgh

PENG DAI, PRAVEEN PARITOSH, and DAVID HUYNH, Google

56

Parsing the semantic structure of a web page is a key component of web information extraction. Successful extraction algorithms usually require large-scale training and evaluation datasets, which are difficult to acquire. Recently, crowdsourcing has proven to be an effective method of collecting large-scale training data in domains that do not require much domain knowledge. For more complex domains, researchers have proposed sophisticated quality control mechanisms to replicate tasks in parallel or sequential ways and then aggregate responses from multiple workers. Conventional annotation integration methods often put more trust in the workers with high historical performance; thus, they are called performance-based methods. Recently, Rzeszotarski and Kittur have demonstrated that behavioral features are also highly correlated with annotation quality in several crowdsourcing applications. In this article, we present a new crowdsourcing system, called Wernicke, to provide annotations for web information extraction. Wernicke collects a wide set of behavioral features and, based on these features, predicts annotation quality for a challenging task domain: annotating web page structure. We evaluate the effectiveness of quality control using behavioral features through a case study where 32 workers annotate 200 Q&A web pages from five popular websites. In doing so, we discover several things: (1) Many behavioral features are significant predictors for crowdsourcing quality. (2) The behavioral-feature-based method outperforms performance-based methods in recall prediction, while performing equally with precision prediction. In addition, using behavioral features is less vulnerable to the cold-start problem, and the corresponding prediction model is more generalizable for predicting recall than precision for cross-website quality analysis. (3) One can effectively combine workers' behavioral information and historical performance information to further reduce prediction errors.

Categories and Subject Descriptors: H.5.m [Information Interfaces and Presentations (e.g., HCI).]

General Terms: Design, Algorithms, Performance

Additional Key Words and Phrases: Crowdsourcing, quality control, behavioral features, worker performance

ACM Reference Format:

Shuguang Han, Peng Dai, Praveen Paritosh, and David Huynh. 2016. Crowdsourcing human annotation on web page structure: Infrastructure design and behavior-based quality control. *ACM Trans. Intell. Syst. Technol.* 7, 4, Article 56 (April 2016), 25 pages.

DOI: <http://dx.doi.org/10.1145/2870649>

1. INTRODUCTION

Since the early days of the web, researchers have endeavored to extract structural information from the billions of web pages. However, accurate extraction is still a challenge due to the high variability of web page designs and the lack of a large-scale labeled dataset for training robust machine-learning algorithms [Chang et al. 2006]. Although there have been several attempts to utilize unsupervised extraction algorithms [Banko

Authors' addresses: S. Han, 135 N Bellefield Ave., University of Pittsburgh, Pittsburgh, PA, 15213; email: shh69@pitt.edu; P. Dai, P. Paritosh, and D. Huynh, 1600 Amphitheatre Parkway, Mountain View, CA 94043; emails: {daipeng, pkp, dfhuynh}@google.com.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

2016 Copyright is held by the owner/author(s). Publication rights licensed to ACM.

ACM 2157-6904/2016/04-ART56 \$15.00

DOI: <http://dx.doi.org/10.1145/2870649>

et al. 2007; Wu and Weld 2010; Song et al. 2010], studies [Wu et al. 2008; Wu and Weld 2010] have found that extraction performance can be significantly improved by incorporating human-labeled information in semisupervised algorithms. This motivates the collection of large-scale training data for supervised and semisupervised web information extraction systems.

Crowdsourcing is an emerging technique for collecting data in a scalable and cost-effective way. It has been used extensively to collect datasets for training machine-learning algorithms. However, due to quality control challenges, the vast majority of existing crowdsourcing tasks, such as those on Amazon Mechanical Turk, are simple, including mostly survey filling, image tagging, and question answering. Web information extraction tasks are usually much less involved since their goal is to mark complex semantic structures within a web page. This drives us to design a new crowdsourcing tool for gathering human labels for web page extraction algorithms. Our tool needs to address the following challenges.

The first challenge is data quality control. Crowdsourcing systems enable task owners to recruit many nonexpert workers. To achieve high quality in the outcome, researchers have developed various quality control strategies mostly based on redundancy. One widely used method is to dispatch each task item to several workers and later apply majority voting on the redundant responses received. More advanced methods, such as weighing annotation quality based on a worker's historical performance [Ipeirotis et al. 2010] or applying Bayesian updates [Dai et al. 2013], are often employed to reduce the degree of redundancy needed. Nevertheless, there is still room for improvement, since any redundant work is productivity loss and extra expense for employers.

One drawback of the historical performance-based method is that it fails to take into account the differences among task items, and that it treats each worker as a consistent workforce on task items of the same type. In reality, workers may perform better on one particular subset of task items than another, which is not caught by these performance-based models. Additionally, workers may occasionally be in different moods or have varying levels of motivation or fatigue. All of these factors would affect their performance. Another limitation is that performance-based methods usually require substantial historical data to build an effective model, therefore easily suffering from the cold-start problem with new workers. One way to overcome these problems is to predict crowdsourcing quality based on worker behavior (e.g., time spent annotating, click and mouseover counts). Indeed, previous work has demonstrated that behavioral features are highly correlated with response quality on multiple crowdsourcing tasks [Rzeszotarski and Kittur 2011, 2012]. Our work is based on these findings but goes further to perform a thorough study of the behavior-based method for our annotation system, using a larger set of behavioral signals.

However, building an annotation system for labeling web page structure and performing necessary data analysis based on the annotation results is a challenge. Annotating web page structure is open ended. The same structure can be annotated differently by different workers because of the lack of a clear boundary for the annotation content. Along with the variability of how a web page is displayed on different devices, at different resolutions, and in different browsers, aligning structure annotations by different workers on the same web page is a challenge. Doing so requires the fuzzy matching of both web page structure information (i.e., the DOM structure) and the corresponding textual information wrapped in the structure. This limits current annotation applications from effectively supporting web page structure annotation [Russell et al. 2008; Stenetorp et al. 2012].

We handle the aforementioned challenges by proposing a new crowdsourcing tool, called Wernicke and implemented as a browser extension, for annotating web page

structure. Wernicke allows users to annotate web page structure as they read a page in their browser. One of Wernicke's important innovations is a new annotation representation schema that decomposes an annotation result as Bag-of-Fragments. This enables downstream quality analysis algorithms to compare annotations by different workers in a standard way (e.g., using standard metrics such as precision and recall). Furthermore, Wernicke provides a novel behavior-based quality control solution to infer annotation quality and resolve conflicts between annotations from multiple workers. This behavior-based method does not require workers' historical annotation information, and thus, is free from the cold-start problem. Finally, the behavior-based method is combined with a conventional-worker-performance-based method, yielding further improvement in the quality of crowdsourcing output.

This article is organized as follows. Section 2 summarizes related work, including techniques of interacting with web page structure and existing quality control strategies for crowdsourcing. Section 3 describes system design rationale and introduces the set of behavioral features. We elaborate on implementation designs, including workflow and behavior logging, in Sections 3.1 to 3.3. To better analyze crowdsourcing output and perform quality control, we then discuss a novel approach to represent, aggregate, and evaluate workers' annotation data in Section 4. We lay out research questions, extensive experiments, and result analysis in Section 5. Section 6 summarizes the main contributions of this article and outlooks for future research directions.

2. RELATED WORK

Crowdsourcing markets like Mechanical Turk provide an inexpensive way to recruit a massive number of workers and quickly collect human-generated data to build large-scale datasets [Kittur et al. 2008; Howe 2006]. However, existing crowdsourcing systems don't support tasks that require sophisticated user interactions. Many new annotation systems have indeed been built for complex tasks (e.g., the BRAT system for complex NLP annotations [Stenetorp et al. 2012] and the LabelMe system for image object labeling [Russell et al. 2008]). However, these systems are specifically tailored to their supported tasks and are not easily reused. To enable web page structural annotation, we need a system that can assist users in identifying and marking web page structure information that consists of both an HTML DOM structure and the textual content wrapped in that structure.

The techniques used for building web page structural annotation systems appear in previous research on the user-guided extraction of semantic information from web pages. Thresher [Hogue and Karger 2005] lets users highlight and label exemplar semantic information in a web page, and then induces HTML patterns to identify other similar HTML elements of the same semantic type(s) on the same page. Sifter [Huynh et al. 2005] and Piggy Bank [Huynh et al. 2006] extract semantic information from web pages and make them readily available for browsing, searching, filtering, and sorting. These two browser extensions are triggered as the user reads a web page. A desirable feature of these extensions is the preservation of the web page's display styles. To build our system, we follow the design rationale of these tools but make several significant enhancements. First, we consolidate the aforementioned features: (1) we design a web browser extension to retain web page display styles, and (2) we provide a simple rule to detect HTML elements of the same type. Second, we add several advanced functions for our crowdsourcing tasks of annotating web page structure. For example, we introduce a new annotation result representation and parsing algorithm, particularly for annotating web page structure, and we also refine several existing quality control mechanisms for producing high-quality crowdsourced data.

Quality control is necessary for crowdsourcing applications because crowdsourcing may suffer from the risk of involving low-quality workers [Dekel and Shamir 2009]. We

categorize the methods to handle such a problem into two categories: (1) the procedure-driven method and (2) the data-driven method.

The procedure-driven approach refers to managing crowdsourcing quality through proper workflow design. Previous studies have developed many strategies, such as designing an iterative workflow to make one worker continue working on another worker's output [Little et al. 2009], decomposing long crowdsourcing tasks into several small and easy-to-achieve subtasks [Bernstein et al. 2010], inserting micro-breaks into long and tedious tasks [Dai et al. 2015], and collecting both a worker's self- and external assessments and using such information for quality evaluation [Downs et al. 2010]. Although proven to be effective, these approaches create complex workflows and often require the tailoring of existing annotation interfaces and procedures. Dai et al. [2013] found that it is possible to provide an automated agent, based on decision-theoretic optimization, for controlling and optimizing a complex workflow. However, in order to apply it, one needs accurate worker performance models, which are hard to acquire.

The data-driven approach does not affect crowdsourcing workflows but requires significant posttask data analysis. The first widely adopted method was proposed by Dawid and Skene [1979]. They developed a simple unsupervised EM algorithm to optimize the estimation of crowdsourcing output quality and worker quality iteratively until convergence. Their method does not require ground-truth information. Once ground truth is available, the quality analysis can be further improved through supervised machine-learning algorithms. The simplest supervised algorithm is to estimate a worker's current crowdsourcing quality by his or her historical performance [Ipeirotis et al. 2010]. This is based on the assumption that each worker can maintain his or her performance at the same level across different tasks and subtasks. However, this is usually untrue, particularly when taking into consideration learning effects, fatigue effects, and task differences. Recently, Rzeszutarski and Kittur [2011] have provided an alternative, behavior-based method for crowdsourcing quality analysis. They found that workers' annotation behaviors (e.g., annotation duration, mouseover, and click activities) on crowdsourcing systems are strong indicators of annotation quality. As a result, they built CrowdScape [Rzeszutarski and Kittur 2012], a crowdsourcing quality control system that displays response quality through interactive visualization of worker behavior and worker output. In this article, we re-explore their approach in a new task setting and consider a more comprehensive set of behavioral features. In particular, we consider lists of novel behavioral features for better inference and evaluation of data quality. Our subsequent studies confirm the utility of behavioral features in measuring annotation quality and, through extensive experiments on quality prediction, produce several interesting findings.

It is worth noting that user behaviors were also widely used in many other research domains. For example, in information retrieval tasks, researchers have found that user behaviors, such as page dwell duration, scrolling, clicking, mouse cursor activities [Huang et al. 2011], and mobile touch interactions [Guo et al. 2013; Han et al. 2015], are highly correlated with users' interests, relevance judgments on web pages, satisfaction, and frustration. Meanwhile, implicit user behaviors have also been adopted in recommender systems to solve the cold-start problem [Hu et al. 2008] or to better model user interests [Yi et al. 2014].

3. CROWDSOURCING SYSTEM DESIGN

This section begins with an introduction of our crowdsourcing system framework in Section 3.1. Then, we elaborate on the task workflow in Section 3.2 and describe the behavior logging function in Section 3.3. Finally, we discuss a list of behavioral features used for crowdsourcing quality prediction in our task (Section 3.4).

Fig. 1. Crowdsourcing system (Wernicke) framework.

3.1. Crowdsourcing System (Wernicke) Framework

We build a system called Wernicke to support web page structural annotation. The overall infrastructure of Wernicke is shown in Figure 1, which consists of a Browser extension module and a quality control module. Web page structural annotation aims to mark up the structure of web pages with correct label(s). Built upon the best practices of existing tools for similar crowdsourcing tasks [Russell et al. 2008; Stenetorp et al. 2012], Wernicke provides support for HTML element selection, highlighting, and labeling. These functions are implemented in the Browser extension.

Because of the importance of quality control for complex crowdsourcing tasks, we conduct an extensive exploration of quality analysis that takes into account both workers' historical performance and their annotation behaviors [Rzeszutarski and Kittur 2012]. To provide proper behavior-based quality control, the Browser extension and the quality control module need to coordinate with each other. Specifically, the Browser extension detects and records annotation behaviors through AJAX requests, and the logged behaviors are extracted and applied in the quality control module.

3.2. Crowdsourcing Workflow in Wernicke

Wernicke's workflow is shown in Figures 2 and 3, which include four steps: (1) A worker installs Wernicke as a web browser extension and triggers it to start an annotation task (Area 1 in Figure 2). (2) After login, Wernicke automatically directs the worker to the first unannotated web page and injects a toolbox (Area 2 in Figure 2). (3) Wernicke assists workers in selecting appropriate HTML elements that correspond to a desired structured content block. Wernicke will highlight any HTML element that a worker covers with the mouse (Area 3 in Figure 2). (4) The worker repeats the previous steps and labels the to-be-annotated HTML elements one by one.

We observe that there are usually multiple to-be-annotated elements in one web page. In Figure 5, for example, a Q&A web page may have multiple comments for the same question. These comments are wrapped in similar HTML elements with minor DOM path differences. Wernicke facilitates a worker's annotation process through an intelligent module that generalizes the annotation of an HTML element to those HTML elements with similar DOM paths; annotating an HTML element triggers the module to detect other similar elements on the same page. Workers can either confirm or cancel the generalized annotation rules (see Area 3 in Figure 3).

Fig. 2. Wernicke annotation workflow on an exemplar Q&A web page.

Fig. 3. An example of Wernicke annotation rule generalization.

3.3. Behavior Logging in Wernicke

To implement behavior-based crowdsourcing quality control, Wernicke provides a behavior logger in the Browser extension. The behavior logger can record workers' crowdsourcing behaviors, including mouse clicking, mouseover, scrolling, key press, page blurring (i.e., losing focus of one page), and page-focusing events. The logged behaviors are used to extract four types of behavioral features that are explained in detail in Section 3.4. Each time a worker triggers any of these events, Wernicke logs the behavior type, the starting and ending time, and the corresponding HTML element and its position (if any). More specifically, a position refers to the horizontal (X) and vertical (Y) distances of the given element from the top left corner of a web page. Wernicke begins logging workers' behaviors after the web page is loaded.

3.4. Behavioral Features

The logged behaviors will be used for extracting behavioral features and further applied for quality analysis. We propose a set of behavioral features under the context of a more generalized type of crowdsourcing task. We refer to this type of task as a web-based

crowdsourcing task—one that can be loaded and completed in a web browser. The goal of this subsection is to provide a general framework for behavioral features that could be applied to other similar crowdsourcing tasks. We categorize these behavioral features into four types. Temporal and page navigation features are directly adopted from previous research [Rzeszutowski and Kittur 2011; Huang et al. 2011]. Contextual and compound features are our newly proposed behavioral features.

- Temporal behavior features** are universally available and simple to compute in any system. In a crowdsourcing task, we argue that there are four generalizable temporal behavior features: (1) total annotation time; (2) before annotation time: duration between when a worker loads a task and annotates any content; (3) after annotation time: duration after a worker annotates the last item but before he or she leaves the web page (after successfully finishing all annotations); and (4) total pausing time: duration a worker remains inactive while performing an annotation, which may indicate “reading” or “thinking” [Guo et al. 2013].
- A worker may navigate around a web page for locating the desired content, particularly when encountering long web pages. We refer to these behaviors as **page navigation behavior features**, which include click, mouseover, and scrolling activities.
- In a web-based crowdsourcing task, annotation content is wrapped in HTML elements, which enables us to track and measure the contextual efforts of labeling an element. For instance, in an annotation task, we compute the total time a worker puts his or her mouse cursor over the annotation content area. These are called **contextual behavior features**.
- It may be difficult for a single behavior to reflect a worker’s intention, which drives us to study two or more behaviors at a time. For example, moving the mouse cursor in and out of an element may indicate that the worker is indecisive, and scrolling up and down several times before annotating may suggest that he or she is evaluating a task. We refer to these types of behaviors as **compound behavior features**.

4. METHODOLOGY FOR QUALITY CONTROL

There are several problems that Wernicke’s quality control module should address. Since web page structural annotation is open ended, the first problem is to develop an approach that makes user annotations comparable to each other. This includes mechanism development for parsing, representing, and matching annotations from different workers. The second problem is to integrate different workers’ annotations to the same web page in order to build a nonconflict resolution. The complete procedure is shown in Figure 4. Individual components in Figure 4 are elaborated in Sections 4.1 to 4.3. After an integrated resolution and the ground-truth information are obtained (Section 4.4), we evaluate the quality of different resolutions generated from different quality control mechanisms (see Section 4.5, where both behavior-based and performance-based quality control methods are considered).

4.1. Representing Annotations as Bag-of-Fragments

We propose a new concept, called Bag-of-Fragments (*BOFs*), to represent a worker’s web page annotation as multiple small, independent fragments. An example of an annotated web page is shown in Figure 5. Each red box denotes the annotation of a fragment and the header of the red box provides the category. Each fragment is represented by worker information, the fragment’s textual content, its DOM path, its display offsets (i.e., the *X*- and *Y*-coordinates in terms of pixels from the top left corner of a web page), and its corresponding category label. The display offsets of the same fragment might be varying under different computer screens and web browser

Fig. 4. An illustration of the resolution generation procedure in Wernicke.

Fig. 5. Bag-of-Fragments (BOFs) representation of an annotation.

settings. Therefore, it is only used in determining contextual behavioral features (see Section 3.4), while it is not used for identifying equivalent fragments from different workers. The labeled category information is used to determine the equivalent fragment in Section 4.2. However, such information is not used for conducting the annotation performance analysis, since we target the labeling process of identifying and annotating all useful fragments of a web page. We do notice quality differences among different categories and plan to study this in future work.

4.2. Identifying Equivalent Fragments from Different Workers

Crowdsourcing tasks usually involve redundancy—a way to distribute multiple replications of the same question to different workers in order to determine the best answer. In a web page annotation task, the same fragment could be annotated slightly differently by different workers because of the high variability of free web page design. To identify the same annotation (i.e., fragment) from different workers, we need to determine the fragment equivalence. In our article, two annotated fragments are judged as equivalent based on the satisfaction of all the following three rules:

- Their category labels are the same.
- Their DOM paths are approximately the same, as defined by two constraints: (1) one DOM path should be totally covered in the other DOM path, and (2) the number

of subpaths in the given two paths should have, at most, two subpath differences. For example, the DOM path “*body > div[1] > H1*” matches with the path “*body > div[1] > H1 > div[1] > H2*,” but it does not match with the path “*body > div[2] > H1*,” since it does not satisfy constraint 1.

—Their texts are adequately similar: the cosine similarity is greater than 0.95.

4.3. Aggregating Annotations from Multiple Workers

To generate a nonconflict resolution for annotations of the same web page from multiple workers, we conduct the following procedure. First, we decompose each annotation of a web page into *BOFs*. Then, a fragment clustering algorithm is employed to create a list of fragment clusters. Each cluster consists of different variants of the same fragment, annotated by different workers. Finally, an output integration algorithm $\mathcal{F}$ is applied to determine whether a fragment cluster should be included in the resolution.

4.3.1. Fragment Clustering. The fragment clustering algorithm attempts to assign each decomposed fragment to its matched fragment cluster. Here, the “matching” between a fragment and a cluster is judged based on whether the given fragment is equivalent (see Section 4.2) to at least one other fragment in the cluster.

The algorithm works in the following way. First, we define a fragment cluster set $\mathbf{S}$ with each element denoting a cluster $\mathbf{c}$. $\mathbf{c}$ consists of multiple equivalent fragments annotated by different workers. We initialize $\mathbf{S}$ to be an empty set. Then, we loop through all fragments within the web page and assign each fragment f into one cluster $\mathbf{c}$. If $\mathbf{S}$ is empty, or f does not match an existing cluster $\mathbf{c}$ in $\mathbf{S}$, a new cluster $\mathbf{c}'$ is created. Then, $\mathbf{c}'$ will be put into $\mathbf{S}$. If f matches an existing cluster $\mathbf{c}$, it will be added into $\mathbf{c}$. As a result, all fragments belong to one cluster and each cluster contains, at most, one fragment per worker.

To examine the accuracy of our fragment clustering algorithm, we conduct an informal evaluation with a random sample of 100 fragment clusters from our data corpus. We manually label whether each fragment within a cluster indeed belongs to the cluster. Among all these sampled clusters, we only detect two questionable clusters, where different fragments are misplaced into the same cluster.

4.3.2. Crowdsourcing Output Integration. Majority voting (MV) is a common method for integrating crowdsourcing output from multiple workers. If one annotation is agreed upon by the majority of workers, it can be resolved as a correct one. This method can be further improved by weighing annotations based on the quality of the contributing worker, which is called weighted majority voting (WMV). Suppose that there are three workers (w_1 , w_2 , and w_3) annotating a web page p , and their corresponding annotation qualities on p are q_1 , q_2 , and q_3 . w_1 and w_2 annotate fragment f , while w_3 does not. In applying WMV, $q_1 + q_2$ is the weighted vote for annotating f and q_3 is the vote for not annotating f . Therefore, we should annotate f if $(q_1 + q_2) > q_3$, and not annotate f otherwise. A detailed description of WMV can be found in Li and Yu [2014]. An odd number of workers is often used for MV, since an even number may easily result in a tie. In this article, we randomly select a side in the case of a tie.

The simplest way to obtain annotation quality is by employing workers’ historical performance. However, the cold-start problem may be encountered when such information is lacking, and this method also cannot differentiate web pages with different annotation complexity. An alternative approach is to measure annotation quality case by case. Previous studies [Rzeszotarski and Kittur 2011, 2012] find that utilizing worker behavior can predict quality in a reasonable way. This drives us to conduct a more comprehensive study with a complete set of behaviors and to further combine behavioral features with the performance-based approach.

Table I. Ground-Truth Fragment Count Using Different #Workers for Consensus*

	#Ground-Truth Fragments		#Ground-Truth Fragments
$N = 1$	4,173 (100%)	$N = 5$	2,036 (48.8%)
$N = 2$	3,081 (73.8%)	$N = 6$	1,735 (41.6%)
$N = 3$	2,747 (65.8%)	$N = 7$	1,414 (33.9%)
$N = 4$	2,313 (55.4%)	$N = 8$	905 (21.7%)

*Under our setting of consensus number ($N = 4$), 55.4% of user-annotated fragments are preserved.

4.4. Ground-Truth and Evaluation Metrics

To evaluate the quality of an annotation or a resolution, we need to know the ground-truth fragments. Such information can be obtained through either expert labeling or a high consensus among multiple workers. In this article, the “correctness” of a fragment is judged based on the consensus among workers: a correct fragment should be annotated by at least N independent workers. The ground-truth fragment is built in the same manner as the generation of resolutions in Figure 5, with the additional step of setting the output aggregation algorithm $\mathcal{F}$ to the high consensus among workers.

Determining N is a tradeoff. A small N may include incorrect fragments, while a large N may filter out good fragments. Following majority voting, we set N to be 4. To understand the quality of the consensus-based ground truth, we manually label 64 randomly sampled fragments and compare the obtained labels with the consensus-based ground truth using the Jaccard Index. The Jaccard Index is 0.7317, which is reasonably high for good data quality. In addition, among the consensus-based fragments, only four disagree with our manual labels. This results in a 93.75% accuracy, with which we can obtain a reliable analysis result.

However, setting $N = 4$ may overexclude several “real” ground-truth fragments. Table I provides ground-truth fragment counts with different consensus number N , where we find that $N = 4$ only includes 55.4% of the user-annotated fragments. Thus, we conduct an additional regression analysis experiment (see Table 5.3 in Section 5.3) based on $N = 3$, where the significance of results stays as $N = 4$ with minor feature coefficient value differences. We think the analysis based on $N = 4$ is robust.

Depending on how an annotation is defined, we can evaluate annotation quality in a couple of ways. First, an annotation could refer to an annotated fragment (e.g., Fragment 2 in Figure 5). Second, it could refer to all of the annotated fragments in a web page (e.g., Fragments 1 through N in Figure 5). The first definition is inappropriate for comparing behavior- and performance-based methods, since not only is the ratio of correct/incorrect fragments highly imbalanced but also it is hard to associate worker behavior on a web page with each particular fragment. This makes the trained machine predictor highly unreliable; therefore, we decide to adopt the second definition. Accordingly, in this article, we evaluate annotation quality based on two standard measures: precision and recall. Precision is defined as the ratio of the number of correctly annotated fragments and the total number of annotated fragments. Recall is defined as the ratio of the number of correctly annotated fragments and the total number of fragments in the ground truth. As an example, for worker w ’s annotation of a web page p , the precision and recall are computed in Equations (1) and (2), where $BOFs(w, p)$ denotes the Bag-of-Fragment representation of w ’s annotation of p and $GT(p)$ denotes the ground-truth fragment clusters for page p :

$$Precision(w, p) = \frac{|BOFs(w, p) \cap GT(p)|}{|BOFs(w, p)|} \quad (1)$$

$$Recall(w, p) = \frac{|BOFs(w, p) \cap GT(p)|}{|GT(p)|}. \quad (2)$$

4.5. Behavior-Based and Performance-Based Quality Control

In this section, we elaborate on the performance-based and behavior-based methods for annotation quality prediction. The performance-based method predicts annotation quality based on a worker's historical performance, while the behavior-based method predicts quality through a linear regression model over multiple behavioral features.

4.5.1. Computing Behavioral Features. In Section 3.4, we classify four types of behavioral features for web-based crowdsourcing tasks. In this section, we define them more specifically within our task context.

Temporal behavior features. The *annotation time* is measured by the time difference between when a given web page is loaded and when its annotation data is submitted. User inactivity time has previously been used to understand whether a user is actually “reading” certain information in web search tasks [Guo et al. 2013]. We borrow this same idea and rename it “pausing.” A pausing behavior is defined as a lack of two consecutive actions within 2 seconds. We compute the total *pausing time* by aggregating all pausing behaviors of a worker's annotation on one web page. For both the pausing and annotation time, we exclude time intervals of two consecutive actions that have more than a 60-second duration between them, which usually indicates that a worker is off-task. These parameters are the same as a previous study [Han et al. 2015]. *Before annotation time* captures the duration of workers' preannotation preparation, which is computed by the time difference between when the web page is loaded and the first annotation action (i.e., the labeling of the first HTML fragment) is conducted. The *after annotation time* is calculated by measuring the time spent after the last annotation action but before the final submission of a given page, which indicates the postannotation reflection time.

Page navigation behavior features. To navigate around a web page, a worker may need to scroll up/down, mouseover/click on HTML elements, and click on Wernicke buttons to add, merge, or remove an annotation on the web page. Page navigation behavior features include the number of scrolls up (*#scrollUp*), scrolls down (*#scrollDown*), mouseovers (*#mouseover*), and clicks (*#click*).

Contextual behavior features. From the *BOF*'s perspective (see Figure 5), annotating is to identify the right fragment and provide a category label for it. We believe workers' behaviors that are directly used to annotate a fragment could be good indicators of annotation quality because they reflect direct annotation effort. These behavioral features are referred to as contextual behavior features. In this article, we consider behavior *b* to be a contextual behavior of fragment *f* if it meets the following requirements: (1) *b* occurs 5 seconds or less before a worker annotates *f*; (2) the current on-operating HTML element (*h*) of *b* is similar to *f*—the subpath difference between their DOM paths should be smaller than four; and (3) the displayed offsets of *h* and *f* are within 100 pixels. If *b* is identified as a contextual behavior for multiple fragments, we choose the one that is annotated immediately after *b* as its corresponding fragment. After obtaining contextual behaviors, we compute the number of clicks (*#conxClick*) and mouseovers (*#conxMouseover*), the total time to annotate these behaviors (*conxAnno time*), and the total pausing time (*conxPausing time*). We do not include contextual scrolling behaviors because of the difficulty in finding their corresponding fragments.

Compound behavior features. When annotating web page structures, workers often scroll up and down and/or move their mouse cursors in and out of an HTML element to locate the minimum fragment that contains all the necessary content. We refer to these as oscillation behaviors and hypothesize that they correlate with annotation quality. Intuitively, more oscillation behaviors may imply that the annotation task is more difficult or the worker is being more careful while annotating. We compute the number of oscillation features for both scrolling and mouseover (i.e., *#oscScroll* and

Table II. Wernicke Behavioral Features*

Feature Type	Features
Temporal behavior feature	annotation time, pausing time , beforeAnno time, afterAnno time
Page navigation behavior feature	#scrollUp, #scrollDown, #mouseover, #click
Contextual behavior feature	conxAnno time , conxPausing time , #conxClick , #conxMouseover
Compound behavior features	#oscScroll , #oscMouseover

*The bolded text indicates proposed new features compared to Rzeszotarski and Kittur [2011].

#oscMouseover). An oscillation behavior refers to two consecutive actions of the same type (scrolling or mouseover): (1) happening within a very short period of time (in our case 2 seconds) and (2) occurring in different directions (both up and down directions for the scroll and both in and out directions for the mouseover). The 2-second threshold is set to be the same as the pausing time threshold. This is to make sure a user is still actively annotating when performing oscillation behaviors.

Note that all the aforementioned parameters are assigned through light-weighted trial and error. Most of the parameters are task dependent, and we do not think that a one-size-fits-all parameter exists for different tasks. We believe that the parameters can be finer tuned to generate even better results.

To summarize, we consider 14 behavioral features under four different categories, as shown in Table II. The bolded text indicates proposed new features compared to Rzeszotarski and Kittur [2011].

4.5.2. Behavior-Based Quality Prediction. The behavior-based method tries to develop a linear regression model from the training dataset. Then, the model is applied to the testing dataset to predict annotation quality. To train the model, the ground-truth precision or recall of a web page annotation is used as the dependent variable, and different behavioral features are used as the independent variables. The goal is to predict precision or recall in the testing dataset. Our subsequent experiments in Section 5.7 show the high utility of successful predictions in generating high-quality annotations.

The regression model is trained based on the following procedure: in the beginning, all 14 behavioral features are included as independent variables in a linear regression model. Then, the AIC-based (i.e., Akaike information criteria) feature selection¹ is applied to select the best regression model with the highest AIC. The selected regression models could slightly vary in different training datasets because of the data differences. Finally, the selected model is employed to predict the quality of each annotation (precision or recall) in the testing datasets. Since the precision and recall both fall in the range of [0,1], the out-of-range predictions are truncated to the boundary values.

4.5.3. Performance-Based Quality Prediction. The performance-based method is employed as a baseline to compare with the behavior-based method. Since the latter does not include any web-page-related features (e.g., web page length or domain) or worker identity information, a fair baseline should also remove such information. Thus, an average-performance-based method, which predicts worker performance based on the average historical performances of all workers, is adopted as our first baseline. Workers' identity information may be available sometimes; thus, we consider a more challenging baseline—the individual-worker-performance-based method, which predicts a worker's future performance based on his or her average historical performance. To ensure proper use of this method, the training and testing datasets in our experiments are specially tailored to contain annotations from all workers.

¹We employ the stepAIC() function in the MASS package of R for model selection. The direction of model selection is set to "both."

After obtaining the predicted annotation qualities based on these two methods, the follow-up problems to be addressed are (1) which prediction is better and (2) whether the predicted qualities can be applied into the crowdsourcing integration algorithm (i.e., $\mathcal{F}$ in Figure 4) to generate high-quality resolutions.

5. EXPERIMENT

5.1. Experiment Overview

This section tries to answer the following five research questions. RQ1: Are the workers' behavior features correlated to annotation quality? RQ2: Are behavior-based methods better than performance-based methods in predicting annotation quality? RQ3: Are behavior-based methods more generalizable? RQ4: Can we integrate behavior-based methods with performance-based methods to further improve prediction accuracy? RQ5: How effective are behavior-based methods in integrating annotations of the same task item from different workers? Note that the annotation quality mentioned in these research questions refers to both precision and recall. To answer these five research questions, we design five experiments.

- We perform regression analysis on different behavioral features, aiming to analyze the relationships between each behavioral feature and annotation quality. This experiment attempts to answer RQ1 and its details are provided in Section 5.3.
- We predict the quality of unseen annotations based on worker behavior. We think the annotation complexity may vary significantly for web pages of different website domains. To eliminate the website domain effect, we maintain the same website distribution in the training and testing datasets. We refer to this as a within-website annotation quality prediction experiment. This experiment attempts to answer RQ2 and its details are presented in Section 5.4.
- As for RQ3, the generalizability is tested based on examining whether the prediction model learned from one website domain can be applied in predicting other website domains. The details of this experiment are presented in Section 5.5.
- We try to improve annotation quality prediction by integrating the behavior-based method with the performance-based method. This experiment attempts to answer RQ4 and its details are presented in Section 5.6.
- To address RQ5, we compare different output integration algorithms described in Section 4.3.2. The details of this experiment are presented in Section 5.7.

5.2. Dataset

We design a user study for crowdsourcing Q&A web page structures and collect a dataset with real workers' annotations using Wernicke. The goal of the user study is to label HTML elements of Q&A web pages and categorize them accordingly, including the "title," "question," "answer," "author," "comment," and "editor." Workers were recruited through a third-party organization. We provide them with annotation instructions, consisting of clearly stated task goals and examples with screenshots and videos. We sample web pages from five Q&A websites. Our sampling algorithm takes into account the content length of a page and samples an equal number of long, medium, and short web pages. Each web page was assigned to eight different workers to ensure enough redundancy for future voting of ground truth.

We evenly divided a total of 32 workers into four groups with eight workers in each. The workers from each group annotated the same batch of 50 web pages, 10 pages from each of the five websites. There were no overlapping web pages across different batches. In total, we have 200 annotated web pages. The workers performed remotely and were paid based on a fixed hourly wage. Our task was assigned on August 11, 2014, and completed on August 12, 2014.

Table III. Regression Analysis of Behavioral Features for Annotation Quality

Feature(s)	Regression Model: $X = P/R/F1$	Precision (P)	Recall (R)	F1
annotation time	$X \sim \text{annotation time}$	-0.0476**	-0.0940**	-0.0788**
pausing time	$X \sim \text{annotation time} + \text{pausing}$	0.0034	0.0136**	0.0110**
beforeAnno time	$X \sim \text{annotation time} + \text{beforeAnno}$	0.0085**	0.0167**	0.0125**
afterAnno time	$X \sim \text{annotation time} + \text{afterAnno}$	-0.0097	0.0206*	0.0154**
#scrollUp	$X \sim \text{annotation time} + \#\text{scrollUp}$	0.0437**	-0.0030	0.0154
#scrollDown	$X \sim \text{annotation time} + \#\text{scrollDown}$	0.0318**	-0.0228*	-0.0111
#mouseover	$X \sim \text{annotation time} + \#\text{mouseover}$	0.0490**	-0.0542**	-0.0244**
#click	$X \sim \text{annotation time} + \#\text{click}$	0.0438**	-0.1274**	-0.0724**
conxAnno time	$X \sim \text{annotation time} + \text{conxAnno}$	0.0357**	-0.0307**	-0.0356**
conxPausing time	$X \sim \text{annotation time} + \text{conxPausing}$	-0.0020	0.0034	0.0015
#conxClick	$X \sim \text{annotation time} + \#\text{conxClick}$	0.0455**	-0.1000**	-0.0503**
#conxMouseover	$X \sim \text{annotation time} + \#\text{conxMouseover}$	0.0297**	-0.0318**	-0.0070
#oscScroll	$X \sim \text{annotation time} + \#\text{oscScroll}$	0.0182	-0.0070	0.0042
#oscMouseover	$X \sim \text{annotation time} + \#\text{oscMouseover}$	0.0236**	-0.0328*	-0.0168

P-Values are based on t-Test. Newly proposed features are bolded. Most behavioral features are significant predictors for annotation quality.

Note *: p-value ≤ 0.05 ; **: p-value ≤ 0.01 .

5.3. Regression Analysis on Behavioral Features

First, we perform a regression analysis to understand whether each behavioral feature is a significant predictor for annotation quality. Since most of the behavioral data are not normally distributed, we use their logarithmic values for smoothing. To avoid the logarithm of zero, each feature value is added by one before log-transformation. Although the transformed feature values are still not normally distributed,² it largely reduces their skewness.³

In regression analysis, the dependent variables are Precision (P), Recall (R), and $F1$ measure (which leverages P and R using the harmonic mean, i.e., $F1 = 2 \cdot P \cdot R / (P + R)$). The independent variables are behavioral features. We first perform a single-feature regression analysis, where we find, quite surprisingly, that all of the behavioral features are negative predictors for annotation quality. We hypothesize that, everything else being equal, having more behaviors may suggest that a web page is more complex. This makes it more difficult to obtain a high-quality annotation. To control annotation difficulty, we train a regression model that keeps the annotation time as a control variable. The results indicate the correlation between behavioral features and annotation precision or recall, given the same amount of time.

Detailed multifeature regression analysis results are provided in Table III, where we highlight the newly proposed behavioral features comparing to previous work [Rzeszutarski and Kittur 2011, 2012]. Most of the newly proposed behavioral features are significant predictors for precision and recall. At the same time, adjusted R-square (goodness of fit) for the regression model increases from 0.2130 (using only nonhighlighted behavioral features) to 0.2288 (using all behavioral features). Table 5.3 also demonstrates the following things: (1) Temporal features, except the annotation time, are all positive predictors for recall. One explanation is that these features reveal workers' diligence level and mental engagement with their work; however, we need further studies for confirmation. (2) Most page navigational, contextual navigation, and compound behaviors are positive predictors of precision but negative predictors of recall.

²We tested it using the Shapiro-Wilk test. The p-values for different behavioral features before and after log-transformation are all smaller than 0.01.

³Skewness values measure how far away the distribution of a feature is from its mean [Joanes and Gill 1998]. The log-transformation reduces skewness values by half and from [2, 4] to [0, 2].

We conjecture that these behaviors indicate that workers are conservative. Again, this hypothesis also requires further research. (3) Since $F1$ leverages both precision and recall, it is interesting to see how the behavioral features with opposite impacts on precision and recall trade off in predicting $F1$. Indeed, both `#scrollDown` and `#conx-Mouseover` no longer exhibit significant impacts on $F1$. To provide more comprehensive understanding of the behavioral features, we report both precision and recall instead of a simple $F1$ measure in our subsequent experiments.

Note that the regression coefficient is different from the correlation coefficient, since the former measures how much the dependent variable changes with a one-unit increase of an independent variable. The regression coefficient is highly related to the range of the feature values. For instance, when predicting annotation precision using the logarithmic-transformed annotation time ranging from (7.82, 13.88), a one-unit increase will result in a precision drop by 4.8% and a recall drop by 9.4%, which is significant. Therefore, a relatively small regression coefficient does not indicate that a feature is of marginal importance. As for the goodness of fit for each regression model, the overall F-test for regression models are all significant at the 0.01 level and the adjusted R-squares are around 0.15.

We also conduct a regression analysis with all 14 behavioral features as independent variables and an AIC-based model selection. Among these 14 behavioral features, the selected regression models for precision (recall) contain eight (nine) of them. The selected features cover all four types. Likewise, almost all of the selected behavioral features are significant predictors of annotation quality ($p\text{-value} \leq 0.05$). Our subsequent experiments on annotation quality prediction are based on the linear regression models of all behavioral features with the same AIC-based feature selection method.

5.4. Within-Website Annotation Quality Prediction

5.4.1. Experiment Setup. Both behavior-based and performance-based methods require a training dataset to gather information about workers' historical performances and train behavior-based regression models (i.e., the coefficients of each behavior feature). Thus, our data is divided into training and testing datasets. To avoid the cold-start problem in performance-based methods, both training and testing datasets contain annotations from all workers.

In this experiment, the training and testing datasets are divided in a way to keep the same distribution of web pages for all websites. A fixed $X\%$ of the web pages (of each worker) from one website are used for training and the rest are used for testing. X remains the same for all websites. The performance of quality prediction depends heavily on the amount of available training datasets. We vary the percentage of data used for training (i.e., $X\%$) to control training data size. In this experiment, we consider nine different values of X (10%, 20%, ..., 90%). We repeat the sampling process 100 times per worker per X value. The reported values are averaged over 100 runs. Since we have 1,600 annotations in total, $X = 0.1, 0.2, \dots, 0.9$ refers to having 160, 320, ..., 1,440 samples for training, respectively.

5.4.2. Evaluation Metrics. Prediction accuracy is evaluated by two standard metrics: MAE (Mean Absolute Error) and $RMSE$ (Root Mean Square Error), which are frequently adopted to measure prediction accuracy in regression analysis [Hyndman and Koehler 2006]. To be specific, for each annotation i , with the predicted performance $f_{pred(i)}$ and the ground-truth performance f_i , MAE and $RMSE$ measure the differences between the predicted value and the ground truth in 1-norm and 2-norm formats, as shown in Equations (3) and (4). Note that the prediction performance refers to both

Fig. 6. *MAE* (left) and *RMSE* (right) evaluations (with S.E.) of within-website prediction of precision (top) and recall (bottom). x-axis: the ratio of data for training; y-axis: mean of *MAE* of *RMSE*. The figures show the better performance of the behavior-based method even without using worker identity information.

precision prediction performance and recall prediction performance:

$$MAE = \frac{1}{N} \sum_{i=1}^N |f_i - f_{pred(i)}| \quad (3)$$

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N |f_i - f_{pred(i)}|^2}. \quad (4)$$

5.4.3. Results Analysis. *MAE* and *RMSE* evaluations of different compared methods are presented in Figure 6, where “Behavior” (in red), “Average_Performance” (in brown), and “Worker_Performance” (in blue) denote the behavior-based method (see Section 4.5.2), the average-performance-based method, and the individual-worker-performance-based method (see Section 4.5.3), respectively. The solid line denotes precision and the dotted line stands for recall.

We have three main findings: (1) The behavior-based method significantly outperforms the average-performance-based method in both precision and recall prediction, which illustrates the effectiveness of the behavior-based method when there is a lack of worker identity information. (2) Even if applying the worker identity information in performance-based methods (i.e., the individual-worker-performance-based method), the behavior-based method is still able to produce significantly smaller error in recall prediction, and keeps the precision prediction error at the same level. (3) Comparing

the predictions for precision and recall, we observe that the behavior-based method works better for predicting recall than precision. This indicates that the behavioral patterns for generating high-recall annotations are relatively normal and easily generalized across different workers or tasks, while it is more difficult to discover patterns for high-precision annotations via workers' behaviors. This may also be closely related to task difficulty or a worker's personal ability. To further explore the reasons for this result, we perform additional studies on cross-website quality prediction, aiming to examine the generalizability of behavioral patterns.

5.5. Cross-Website Annotation Quality Prediction

5.5.1. Experiment Setup. In this experiment, the training and testing datasets are divided by websites: annotations from some websites are exclusively used for training, while the rest are used for testing. To be specific, we use all of the web pages from website set D to predict annotation quality of pages from a different website set D' . We treat the number of training websites (i.e., $|D|$) as one parameter, which ranges from 1 to 4 since we have five different websites. $|D| = 1$ means that we use workers' annotations from one website to predict annotation quality for the remaining four. Note that we have five, 10, 10, and five different divisions for $|D| = 1, 2, 3, 4$; the reported results are the mean values. The evaluation metrics of cross-website prediction are still the same as the within-website prediction, where we also adopt the *MAE* and *RMSE*.

5.5.2. Result Analysis. *MAE* and *RMSE* evaluations of the three compared methods are provided in Figure 7. Just as with the within-website quality prediction, we observe that the behavior-based method is superior to the performance-based method in recall prediction, while there is no obvious difference in precision prediction. The findings remain stable even when using a different amount of training data (i.e., the results hold for $|D| = 1, 2, 3$, and 4). It is also worth noting that the average-performance-based method and the individual-worker-performance-based method have less difference than they do in within-domain prediction, which suggest that website difference is more dominating than worker difference.

5.5.3. Within-Website Prediction Versus Cross-Website Prediction. Due to the annotation variability of different websites, we hypothesize that cross-website prediction is less powerful than within-website prediction. Thus, we study the prediction consistency of the behavior-based method and performance-based method in this section. We choose the individual-performance-based method because it has similar prediction power as the average-performance-based method in cross-website prediction, but performs better in within-website prediction. To make a fair comparison, we assign equivalent numbers of training and testing web pages to cross-website and within-website training. For example, in cross-website prediction, training with one website is equivalent to using a training ratio of 20% in within-website prediction. Similarly, training with two, three, and four websites is equivalent to training ratios of 40%, 60%, and 80% accordingly.

We define the value changes from cross-website prediction to within-website prediction as ΔC (cross-website minus within-website). C can be either *MAE* (*RMSE*) of the predicted precision based on a worker's historical precision (Performance_P); the predicted recall based on historical recall (Performance_R); predicted precision based on behaviors (Behavior_P); or predicted recall based on behaviors (Behavior_R). A positive ΔC means that the cross-website prediction has greater error. The comparison results are provided in Table IV. We find that all ΔC are positive and most of the differences are significant, which confirms our hypothesis that cross-website prediction produces relatively larger errors.

Fig. 7. *MAE* (left) and *RMSE* (right) evaluations (with standard error) of cross-website prediction of precision (top) and recall (bottom). x-axis: the ratio of data for training; y-axis: mean of *MAE* of *RMSE*. The figure shows the better performance of the behavior-based method in predicting recall, while keeping the same performance level in precision prediction.

Table IV. *MAE* (Top Value of Each Cell) and *RMSE* (Bottom Value of Each Cell) Changes (i.e., ΔC) from Cross-Website to Within-Website Prediction

Training Ratio	Δ Performance_P	Δ Behavior_P	Δ Performance_R	Δ Behavior_R
20%/1 site (~320 samples)	0.026	0.030	0.029**	0.027**
	0.049**	0.032**	0.050**	0.039*
40%/2 sites (~640 samples)	0.019**	0.027**	0.022**	0.019**
	0.029**	0.030**	0.029**	0.023**
60%/3 sites (~960 samples)	0.018**	0.027**	0.019**	0.015
	0.022	0.025*	0.023**	0.014*
80%/4 sites (~1,280 samples)	0.016	0.024*	0.019**	0.015
	0.012	0.019	0.019*	0.011

Mann-Whitney U test is used for testing whether ΔC is significant. Behavior-based method is more generalizable in recall prediction (smaller ΔC) and performance-based method is more generalizable in precision prediction.

Note *: p-value ≤ 0.05 ; **: p-value ≤ 0.01 .

When comparing the behavior-based and performance-based methods, we find that ΔC has different trends in predicting precision and recall. The behavior-based method has bigger ΔC than the performance-based method for precision, while it has smaller ΔC for recall. In other words, the worker performance method for precision is more generalizable than the identified behavior patterns for precision, while the reverse is true for recall. This is an interesting phenomenon. Although the underlying reasons for this are still unclear, there are a couple possible explanations: (1) precision measures

a worker's individual ability to identify correct answers, which is not easily measured purely by behaviors, and (2) high-recall annotations usually have a large number of richer behaviors, which enable us to better understand the behavioral patterns.

5.6. Combining Performance-Based Method with Behavior-Based Method

5.6.1. Experiment Setup. Observing that both performance-based and behavior-based methods help predict precision and recall, we further examine the possibility of combining them for additional improvement of the prediction performance. Since within-website prediction is, in general, better than cross-website prediction, we use the within-website training-testing division for all of the subsequent experiments. In addition, we adopt the individual-worker-performance-based method instead of the average-performance-based method because of its better performance.

The better prediction accuracy of the individual-worker-performance-based method over the average-performance-based method relies on the fact that the former takes into account worker identity information. To integrate behavioral information with worker identity information, we examine the following three integration models: (1) Model I: trains a behavioral-feature-based regression model per each worker; (2) Model II: trains a behavioral-feature-based regression model by adding worker identity as an additional indicator variable; and (3) Model III: trains a behavioral-feature-based regression model by adding workers' historical performances (both precision and recall) as independent variables. We find that Model I performs the worst, probably due to insufficient training data per worker. Model III performs better than Model II because it takes into account both worker identity information and historical performance. In Model II, (*MAE*, *RMSE*) are (0.1877, 0.2536) for precision and (0.1938, 0.2472) for recall, while Model III reduces errors to (0.1817, 0.2491) for precision and (0.1932, 0.2457) for recall. As a result, our subsequent experiments are based on the results of using Model III. All these values are reported based on within-website prediction with training ratio 0.5.

5.6.2. Results Analysis. *MAE* and *RMSE* evaluations of precision and recall predictions are shown in Figure 8, where "Behavior," "Worker_Performance," and "Combined" denote the behavior-based method, the individual-worker-performance-based method, and the combined method, respectively. We observe that the simple feature combination method using the linear regression model is sufficient to yield a significantly better performance than using the two methods separately. The results remain true for both precision prediction and recall prediction, indicating that the behavioral features and performance features contain complementary signals about crowdsourcing quality. However, we also observe that the prediction performance in the combined model may depend on the training data size. In order to produce better performance for the pure behavior-based method, more than 20% of the dataset is required for training for our dataset. In our dataset, 20% equals 320 annotations from 32 workers. The specific number of annotations or percentage of annotations for producing reasonable performance may vary, depending on the quality of data and the annotation tasks.

5.7. Crowdsourcing Output Aggregation

Crowdsourcing tasks usually distribute replicated task items to multiple workers so that the annotations from different workers can be integrated through voting. Commonly used crowdsourcing output integration methods include majority voting and weighted majority voting (see Section 4.3.2).

5.7.1. Experiment Setup. In this section, we compare five crowdsourcing output integration methods.

Fig. 8. *MAE* (left) and *RMSE* (right) evaluations (with standard error) on the precision prediction (top) and recall prediction (bottom). x-axis: the ratio of data for training; y-axis: mean of *MAE* of *RMSE*. The figure suggests better performance of the combined method than both the behavior-based and performance-based method.

- MV refers to the majority voting method that assumes all workers can generate annotations of the same quality. We randomly select one side in the case of a tie.
- WMVP uses the individual-worker-performance-based method (Section 4.5.3) to predict annotation quality. Then, the quality score is used for weighting and voting annotations. Note that the average-performance-based method (Section 4.5.3) produces the same results as MV because it predicts the quality of each worker to be the same.
- WMVB is similar to WMVP, except that it uses the behavior-based method to predict annotation quality (Section 4.5.2) in WMV.
- WMVC uses the combined method (Section 5.6) to predict annotation quality in WMV.
- WMVO uses the “oracle” annotation quality to weight annotations in WMV. The “oracle” annotation quality is directly computed based on the ground truth. The goal is to set an upper of performance for listed methods.

WMV-based methods employ predicted annotation quality, through the behavior-based or performance-based approach, for crowdsourcing output integration. In this study, the annotation quality refers to precision, since it meets the main objective of MV algorithms—to find the correct answers. To predict annotation quality, we require training data. In order to guarantee that we can simulate different redundancy numbers in our subsequent experiments, all eight workers’ annotations of one web page are put together, either in training datasets or testing datasets. Similar to

Fig. 9. The precision (y-axis) of generated resolutions for different crowdsourcing integration methods under different #workers (left) and different training ratios (x-axis) with two workers (right). This demonstrates the usefulness of utilizing predicted quality with both the performance-based and behavior-based method to generate reliable resolutions. These two methods can be combined together to further improve performance.

within-website sampling, we also keep the same distribution of web pages from all websites. Specifically, for each website, we sample a fixed $X\%$ of web pages for training and the remaining for testing. All workers' annotations of the training web pages are used for model training, and all annotations for the rest of the web pages are for testing. Additionally, we use nine values of X (10%, 20%, ..., 90%) to control the amount of training data.

5.7.2. Result Analysis. Our first experiment studies the impact of the number of workers on crowdsourcing output integration performance. We set the training ratio (X) to be 0.7 for the following reasons: (1) As shown in Figures 6 and 8, a large training ratio is preferred for smaller prediction errors. On the other hand, variance rises when the training ratio increases. Therefore, a middle-range training ratio helps minimize both error and variance. (2) We observe, in Figure 8, that the biggest MAE and RMSE differences between the combined method and the two single-feature-based methods occur when $X = 0.7$.

Experimental results are shown on the left side of Figure 9. As expected, we observe different trends for redundancy with odd and even numbers, which is partly due to the random tie-breaking strategy. We also observe the following results: (1) The "oracle"-quality-based WMV (i.e., WMVO) performs the best among all five methods, indicating the necessity of predicting annotation quality beforehand. This also accredits our effort of extensively exploring behavior-based and performance-based methods in predicting the "oracle" quality in the previous sections. WMVO sets the upper bound of the output integration methods. (2) WMV-based output integration methods, including WMVB, WMVP, and WMVC, perform better than pure MV, particularly for an even number of workers. This again demonstrates the utility of our quality prediction algorithms. However, we do not observe vast differences between these three methods, although WMVC generates slightly better results than the others for two workers. (3) On the other hand, we also observe that, with odd numbers of workers, MV performs comparably well to WMVB, WMVP, and WMVC. When both worker behavior and identity information are unavailable, setting crowdsourcing redundancy to be an odd number can help achieve a reasonably good performance.

We then fix the replication number to two and vary the training ratio. We choose an even-number replication as it is a hard case to deal with in real applications. The experimental results are presented on the right side of Figure 9, which shows

comparable conclusions that we drew from the first experiment. In addition, similar to Figure 6, the performance-based method performs slightly better than the behavior-based method. Additionally, the combined method performs better than the other two methods. The Wilcoxon sign test shows that the combined method is significantly better than the behavior-based method on training ratios 0.3, 0.6, 0.7, and 0.8, while we only find a marginal significance ($p\text{-value} = 0.09$) on training ratio 0.7 when comparing it to the performance-based method. Although we observe the combined method to be the best among all three methods in Figure 8, we cannot effectively convert it into a performance boost in output aggregation.

Overall, the experimental results suggest that the behavior-based method can provide significant improvement for crowdsourcing output integration when workers' historical performance information is unavailable (i.e., comparing WMVB to MV). However, we do not observe much improvement when factoring in the behavior-based features. We think this is due to the following reasons. First, our task is relatively simple and a single worker has already achieved relatively high precision, which is difficult to improve upon. Second, annotation quality is measured by predicted precision. As we learned in Sections 5.3 to 5.5, the behavior-based method is more powerful in predicting recall than precision. Third, our prediction algorithm is by no means optimal, so there is still space for a more advanced prediction model.

5.8. Summary of Findings

We perform extensive experiments for understanding the utility of the behavior-based method in predicting annotation quality and integrating crowdsourcing output from multiple workers. We summarize our answers to the research questions raised in Section 5.1.

RQ1 is answered in the regression analysis over a list of behavioral features in Section 5.3, where we find that most behavioral features are correlated with annotation quality. This is because different behavioral patterns reflect workers' engagement with the task. For example, the feature "after annotation time" is a significant indicator of recall, even if it means that workers are inactive. This deserves further study.

In comparing the prediction accuracy between the behavior-based and performance-based methods, we find that behavior-based methods significantly outperform performance-based methods in recall prediction while not sacrificing precision prediction. The results remain true for both within-website prediction and cross-website prediction, demonstrating that behavior-based methods can be generalized across website domains. A comparison between cross-website prediction and within-website prediction reveals that the error of recall prediction changes less in the behavior-based method and more in precision prediction. This suggests that behavior-based methods are more generalizable for recall prediction. Another benefit of behavior-based methods is that they do not require worker identity information, which shows they are even more generalizable across workers and less vulnerable to the cold-start problem. These findings answer RQ2 and RQ3.

We additionally find that behavior-based methods can be effectively integrated with performance-based methods to further reduce prediction errors, not only for recall but also for precision. This answers RQ4.

Regarding RQ5, we show that behavior-based methods can improve crowdsourcing output integration accuracy against majority voting without utilizing workers' historical performance information. However, when integrating behavioral signals and historical performance signals, we do not observe a significant improvement as we expected. This deserves further study.

6. CONCLUSIONS AND FUTURE WORK

6.1. Conclusions

In this article, we conduct a systematic study to understand the feasibility and effectiveness of inferring crowdsourcing quality based on workers' behavioral signals. This work makes several contributions.

First, through extensive experiments, we demonstrate that a quality-control model using workers' behavioral signals is more effective than using workers' historical performance signals in several aspects: (1) it produces significantly better recall prediction without sacrificing precision, (2) it does not rely on workers' historical or individuals' behavioral information and thus generalizes better across websites and workers, (3) it can be combined with workers' historical performance signals to further reduce prediction error for both precision and recall, and (4) it can be used to integrate crowdsourcing output, using weighted majority voting, which achieves higher precision.

Second, we design and implement Wernicke, a crowdsourcing system that supports web page structural annotation. Wernicke provides various crowdsourcing output integration solutions, including a performance-based method, a behavior-based method, and a combined method, for generating high-quality crowdsourcing output.

Third, we propose the Bag-of-Fragments (*BOFs*) concept to represent, parse, compare, and integrate crowdsourcing output consisting of multiple fragments/blocks of user-generated annotations. *BOFs* can be used in other crowdsourcing tasks, such as labeling for NLP tasks and fine-grained image annotation for image object recognition.

6.2. Future Work

Our work opens the door for several future research directions.

First, despite the superiority of behavior-based approaches over the performance-based methods in annotation quality prediction, there is still space to better utilize behavioral information at output integration. In the future, we plan to try more advanced quality prediction models and crowdsourcing output integration algorithms.

Second, we focus on the task of annotating Q&A web pages in this article. However, we believe that Wernicke is a very general framework and want to explore more. Wernicke can be used to assist other web information extraction tasks, such as detecting boilerplate content (e.g., navigational elements, templates, and advertisements) of a web page [Kohlschütter et al. 2010]. Additionally, behavior-based logging and quality prediction approaches developed in Wernicke are useful for traditional crowdsourcing tasks, such as online survey answering and image labeling. The Bag-of-Fragments representation of annotation results can be applied to other crowdsourcing tasks, such as part-of-speech tagging, when each annotated word is viewed as a fragment.

Third, we plan to extend our work to build a more fine-grained, behavior-based quality prediction algorithm for inferring annotation quality of individual fragments. We plan to study whether utilizing contextual behaviors for a specific annotated fragment can predict fragment-level annotation quality.

REFERENCES

- Michele Banko, Michael J. Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. 2007. Open information extraction for the web. In *International Joint Conference on Artificial Intelligence*, Vol. 7. 2670–2676.
- Michael S. Bernstein, Greg Little, Robert C. Miller, Björn Hartmann, Mark S. Ackerman, David R. Karger, David Crowell, and Katrina Panovich. 2010. Soylent: A word processor with a crowd inside. In *Proceedings of the 23rd Annual ACM Symposium on User Interface Software and Technology*. ACM, 313–322.
- Chia Hui Chang, Mohammed Kayed, Moheb R. Girgis, and Khaled F. Shaalan. 2006. A survey of web information extraction systems. *IEEE Transactions on Knowledge and Data Engineering* 18, 10 (2006), 1411–1428.

- Peng Dai, Christopher H. Lin, and Daniel S. Weld. 2013. POMDP-based control of workflows for crowdsourcing. *Artificial Intelligence* 202 (2013), 52–85.
- Peng Dai, Jeffrey Rzeszutarski, Praveen Paritosh, and Ed Chi. 2015. And now for something completely different: Improving crowdsourcing workflows with micro-diversions. In *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW'15)*. ACM.
- Alexander Philip Dawid and Allan M. Skene. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Applied Statistics* 28, 1 (1979), 20–28.
- Ofar Dekel and Ohad Shamir. 2009. Vox populi: Collecting high-quality labels from a crowd. In *Proceedings of the 22nd Annual Conference on Learning Theory*.
- Julie S. Downs, Mandy B. Holbrook, Steve Sheng, and Lorrie Faith Cranor. 2010. Are your participants gaming the system? Screening mechanical turk workers. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2399–2402.
- Qi Guo, Haojian Jin, Dmitry Lagun, Shuai Yuan, and Eugene Agichtein. 2013. Mining touch interaction data on mobile devices to predict web search result relevance. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 153–162.
- Shuguang Han, Zhen Yue, and Daqing He. 2015. Understanding and supporting cross-device web search for exploratory tasks with mobile touch interactions. *ACM Transactions on Information Systems (TOIS)* 33, 4 (2015), 16.
- Andrew Hogue and David Karger. 2005. Thresher: Automating the unwrapping of semantic content from the world wide web. In *Proceedings of the 14th International Conference on World Wide Web*. ACM, 86–95.
- Jeff Howe. 2006. The rise of crowdsourcing. *Wired Magazine* 14, 6 (2006), 1–4.
- Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 8th IEEE International Conference on Data Mining, 2008 (ICDM'08)*. IEEE, 263–272.
- Jeff Huang, Ryan W. White, and Susan Dumais. 2011. No clicks, no problem: Using cursor movements to understand and improve search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1225–1234.
- David Huynh, Stefano Mazzocchi, and David Karger. 2005. Piggy bank: Experience the semantic web inside your web browser. In *The Semantic Web (ISWC'05)*. Springer, 413–430.
- David F. Huynh, Robert C. Miller, and David R. Karger. 2006. Enabling web browsers to augment web sites' filtering and sorting functionalities. In *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology*. ACM, 125–134.
- Rob J. Hyndman and Anne B. Koehler. 2006. Another look at measures of forecast accuracy. *International Journal of Forecasting* 22, 4 (2006), 679–688.
- Panagiotis G. Ipeirotis, Foster Provost, and Jing Wang. 2010. Quality management on Amazon mechanical turk. In *Proceedings of the ACM SIGKDD Workshop on Human Computation*. ACM, 64–67.
- D. N. Joanes and C. A. Gill. 1998. Comparing measures of sample skewness and kurtosis. *Journal of the Royal Statistical Society: Series D (The Statistician)* 47, 1 (1998), 183–189.
- Aniket Kittur, Ed H. Chi, and Bongwon Suh. 2008. Crowdsourcing user studies with mechanical turk. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 453–456.
- Christian Kohlschütter, Peter Fankhauser, and Wolfgang Nejdl. 2010. Boilerplate detection using shallow text features. In *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining*. ACM, 441–450.
- Hongwei Li and Bin Yu. 2014. Error rate bounds and iterative weighted majority voting for crowdsourcing. *arXiv preprint arXiv:1411.4086* (2014).
- Greg Little, Lydia B. Chilton, Max Goldman, and Robert C. Miller. 2009. TurkIt: Tools for iterative tasks on mechanical turk. In *Proceedings of the ACM SIGKDD Workshop on Human Computation*. ACM, 29–30.
- Bryan C. Russell, Antonio Torralba, Kevin P. Murphy, and William T. Freeman. 2008. LabelMe: A database and web-based tool for image annotation. *International Journal of Computer Vision* 77, 1–3 (2008), 157–173.
- Jeffrey Rzeszutarski and Aniket Kittur. 2012. CrowdScape: Interactively visualizing user behavior and output. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*. ACM, 55–62.
- Jeffrey M. Rzeszutarski and Aniket Kittur. 2011. Instrumenting the crowd: Using implicit behavioral measures to predict task performance. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*. ACM, 13–22.
- Xinying Song, Jing Liu, Yunbo Cao, Chin-Yew Lin, and Hsiao-Wuen Hon. 2010. Automatic extraction of web data records containing user-generated content. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*. ACM, 39–48.

- Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. BRAT: A web-based tool for NLP-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 102–107.
- Fei Wu, Raphael Hoffmann, and Daniel S. Weld. 2008. Information extraction from wikipedia: Moving down the long tail. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 731–739.
- Fei Wu and Daniel S. Weld. 2010. Open information extraction using wikipedia. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 118–127.
- Xing Yi, Liangjie Hong, Erheng Zhong, Nanthan Nan Liu, and Suju Rajan. 2014. Beyond clicks: Dwell time for personalization. In *Proceedings of the 8th ACM Conference on Recommender systems*. ACM, 113–120.

Received February 2015; revised December 2015; accepted December 2015