九州大学学術情報リポジトリ
Kyushu University Institutional Repository

Automating the Choice Between Single or Dual Annotation for Classifier Training

Fukuda, Satoshi
Chuo University

Ishita, Emi
Kyushu University

Tomiura, Yoichi
Kyushu University

Oard, Douglas W.
University of Maryland

https://hdl.handle.net/2324/4794468

出版情報：Lecture Notes in Computer Science. 13133, pp.233-248, 2021-11-30. Springer Nature
バージョン：
権利関係：

Automating the Choice Between Single or Dual Annotation for Classifier Training

Satoshi Fukuda[1], Emi Ishita[2], Yoichi Tomiura[2], and Douglas W. Oard[3]

[1] Chuo University, Tokyo 112-8551, Japan
[2] Kyushu University, Fukuoka 819-0395, Japan
[3] University of Maryland, College Park, MD 20742, USA
sfukuda277@g.chuo-u.ac.jp

Abstract. Many emerging digital library applications rely on automated classifiers that are trained using manually assigned labels. Accurately labeling training data for text classification requires either highly trained coders or multiple annotations, either of which can be costly. Previous studies have shown that there is a quality-quantity trade-off for this labeling process, and the optimal balance between quality and quantity varies depending on the annotation task. In this paper, we present a method that learns to choose between higher-quality annotation that results from dual annotation and higher-quantity annotation that results from the use of a single annotator per item. We demonstrate the effectiveness of this approach through an experiment in which a binary classifier is constructed for assigning human value categories to sentences in newspaper editorials.

Keywords: Text classification, Efficient annotation, Multi-armed bandits.

1 Introduction

As a result of open data promotion, various types of data have become available through data repositories and data archives. New types of research, for example, digital humanities or computational social science, are greatly benefiting from them. Many studies that use digital data and text have been conducted using machines. By contrast, qualitative research is still useful because it can provide deeper knowledges than the insights. However, qualitative research requires human effort; hence, the amount of text and data that can be analyzed is limited. We are interested in introducing machines to qualitative research methods while maintaining the quality of thought and improving efficiency. This would enable us to support digital humanities and computational social sciences. Moreover, it would increase the importance of digital archives and data repositories.

Supervised machine learning is widely used for text classification in tasks that range from academic studies of collaboration behavior to commercial tracking of brand mentions on social media. To construct good classifiers, substantial quantities of annotated training data are typically required. Human coders are typically used to create such training data, but this choice results in limits to the amount of annotation

that can be performed for cost reasons. Therefore, we are interested in techniques to create the most useful annotated training data for a given annotation cost.

For manual annotation, Ishita et al. explored two approaches, each assuming that two coders were available [11]. One approach had coders working alone to annotate the largest possible number of different texts for a given level of annotation effort, and the other had coders working together to create the best possible annotation for each text, albeit at a higher cost per annotation. Ishita et al. compared the classifiers trained using each approach by plotting learning curves to investigate how well the classifiers performed as the number of annotations increased for six annotation tasks, and found three broad categories of patterns: (1) rapidly generating large quantities of annotations was sometimes a good approach, particularly when positive examples were rare, (2) generating higher quality annotations by having two coders annotate each text independently and then discussing their disagreements was also sometimes a good approach, particularly when the task was difficult, and (3) starting out with single annotation but switching to dual annotation at some point in the process was sometimes the best overall approach. Clearly, it may also be the case that one coder is better than another, and because of learning and fatigue effects, such differences may change over time. Therefore, in this paper, we regard the choice between a single annotator and dual annotation, and for single annotation the choice of the coder, as an optimization problem, and present a method that consistently makes near-optimal choices over a range of tasks.

If the outcome of each choice could be known with certainty, it would be best to select the pair of a specific coder and a specific text that yields the greatest possible improvement in the performance of the classifier after the next annotation. Clearly, we do not know which label would actually be assigned in each case, but despite this, we can compute the expected score of this improvement using the results of past annotations. Two types of annotations are possible when two coders are available: labeling an unannotated text to increase the quantity of training data or improving an existing label as a result of adjudication (i.e., discussion and joint decision) between coders after obtaining a second annotation for a text that has already been annotated once. If the labeling of unannotated text by a coder improves the performance of the classifier, then the expectation that this coding method will be further improved in terms of classifier performance in the next annotation will be higher. If, by contrast, the expectation of a second annotation that results from adjudication improves the performance of the classifier as the annotation progresses and its expected score is higher than that of a new annotation action, a higher performance classifier will be constructed when an adjudication action is selected for the next annotation.

The sequential choice of the coding method to obtain a higher-performance classifier is modeled in the framework of a multi-armed bandit problem, which is a problem setting for reinforcement learning. Reinforcement learning is a learning approach that seeks to learn to act in a manner that maximizes a reward. In the multi-armed bandit problem, the actions are choices (called “arms”) whose past probability of success can be observed, and the goal is to develop a policy for sequentially selecting the best

choice at each point in time to maximize the total reward over a finite time.[1] For our goal of affordably building the most useful training data, the size of the reward is the quality of the resulting classifier, and the choices are between single or dual annotation. For comparability with prior results, we tested our approach using the datasets constructed by Ishita et al. [11], and found that our approach consistently made near-optimal choices. From these results, we conclude that a similar multi-armed bandit approach to choosing between single or dual annotation could be useful for a broad range of applications in which text classifiers must be trained in a cost-effective manner.

2 Related Work

In this section, we describe prior work on efficiently constructing training data and multi-armed bandits.

2.1 Efficient Construction of Training Data

Several information retrieval evaluation organizations (e.g., TREC and NTCIR) have created large-scale annotated datasets using pooling [12, 17]. In pooling, the annotation task is partitioned by topic, with a single coder judging all texts found above some rank cutoff for that topic using any system that contributed to the creation of the pool. This approach models a case in which the appropriate annotation is an individual opinion, as is typically the case for relevance judgements in a search task. Although this is an efficient approach, in contrast to personal opinions about search topics, our interest is in the annotation of phenomena on which people can agree.

In pooling, machines work alone to select items for annotation. Other approaches are possible, including active learning in which humans and machines work together to determine which documents should be annotated. For example, Cai et al. [4] proposed an active learning method to automatically select the text to be labeled next from an unlabeled dataset using the already annotated items. In this method, the non-annotated text furthest from the already annotated items is chosen to be labeled next. Somewhat closer in spirit to our proposed approach, Culotta and McCallum [7] proposed a method called expected model change maximization to determine the text to be labeled next, which is the text that is expected to most change the discriminative model after the new annotation is added to the labeled dataset, comparing the discriminative model generated from the already-labeled dataset. The distinction between their study and our approach is that they proposed choosing the item to be annotated, whereas we propose choosing the coder who will perform the annotation. Both are important problems, and in future work, the two should be explored together. However, for the initial study reported in this paper, considerable simplification results from our choice to focus simply on the coder selection problem.

[1] The name comes from a colloquial reference to slot machines used by gamblers called "one-armed bandits." In the imagined multi-armed bandit scenario, the gambler seeks to pull the arm that would yield the greatest profit.

All the aforementioned approaches focus on a single annotator. However, we are interested in annotation tasks in which the phenomenon to be annotated is defined by consensus rather than an individual opinion. To evaluate the quality of the annotations between two coders, κ [5, 6], *S* [3], and π [15] coefficients have been widely used. For example, Artstein and Poesio [1] and Fort et al. [8] verified the reliability of the agreement between two coders using various coefficients. When using coefficients to measure the reliability of annotations between coders, it is necessary to conduct the annotation on some fixed collection that both coders annotate. We use such a collection in our experiments, but our goal is to determine optimal choices that can limit dual annotation to those cases in which it is actually adding value.

In recent years, the use of many annotators has been widely studied in the context of crowdsourcing [10], where it is possible to request annotations from a broad range of coders. A number of frameworks for obtaining useful results from crowdworkers who exhibit varying levels of expertise, ability, and diligence have been studied. For example, Nguyen et al. [13] proposed a method to measure the quality of annotations obtained from crowdworkers using a dataset labeled by domain experts using active learning. Welinder et al. [18] proposed a method for estimating true labels by modeling the worker's answering process. Zhang et al. [19] proposed a framework to manage the quality of coders by clustering coders based on their annotation history. However, crowdworkers normally operate independently, whereas we are interested in settings in which disagreeing coders can meet to resolve their disagreements.

2.2 Multi-Armed Bandit Problem

Many algorithms have been proposed for the bandit problem: ε-greedy algorithm, UCB (Upper Confidence Bound) [2], and Thompson sampling [16] are typical algorithms. The ε-greedy algorithm is a method in which an arm is randomly selected with a probability of ε ("exploration") and the arm with the highest expected score for the reward at that time is selected with a probability of $1 - \varepsilon$ ("exploitation"). By considering the balance between "exploration" and "exploitation" for the arm selection, the risk of continuing to select one arm can be mitigated. UCB is a method for calculating the score of each arm using the average of the observed reward and the value calculated based on the number of times the arm is selected. In this method, the lower the number of times the arm is selected, the higher the score. Thompson sampling is a method for selecting an arm based on the probability that the arm can obtain the maximum expected score. In the method, the posterior distribution of the expected score of the reward is modeled in the framework of Bayesian statistics. In a general bandit problem, rewards follow the Bernoulli distribution; that is rewards were obtained or not obtained. By contrast, in the construction of training data, elaborate rewards for the annotation action can be measured by calculating the differences of the performance between the current classifier and previously constructed the classifier. We therefore apply the ε-greedy algorithm, which is the simplest of the bandit algorithms, which can flexibly incorporate the above element.

The above bandit strategies assume that the reward acquisition probability of each arm does not change. However, in reality, the reward distribution of the arm can

change over time (e.g., changes in the click rate for the advertisements caused by changes in ad design and trends). The bandit problem corresponding to such a temporal change of the reward distribution is called a non-stationary multi-armed bandit problem. Discounted UCB [9] and discounted Thompson sampling [14] are the algorithms used to apply UCB and Thompson sampling to this problem, respectively, and they introduce the concept of the discount. In the annotation for the construction of the training data, past and present scenarios may have changed the nature of the labeled dataset and coders. Therefore, by introducing a decay function when evaluating the arm, we expect that an arm with a high expected score can be selected effectively while the influence of past observation results is diminished.

3 Selection Algorithm

3.1 Overview

We are interested in binary annotation tasks in which some phenomenon (in our case, a consensus definition of topical relevance) is to be annotated as present or absent. Our goal is to maximize some single-valued measure of classification effectiveness, such as F1. Assuming, without loss of generality, that there are two coders, and assuming, as a computational simplification, that the order of texts to be annotated is fixed, there are four choices to be made about which coder annotates which text:

- Arm 1: Select coder 1 and the first text in the unannotated set.
- Arm 2: Select coder 2 and the first text in the unannotated set.
- Arm 3: Select coder 2 and the first text in the labeled text set annotated by coder 1, and decide the final annotation after discussion.
- Arm 4: Select coder 1 and the first text in the labeled text set annotated by coder 2, and decide the final annotation after discussion.

For the first two arms, the selected unannotated text is annotated by coder 1 or 2. For the last two arms, coder 1 or 2 annotates the selected text; the two coders discuss which annotation is correct, if their annotations are different; and then the final annotation is decided.

For the calculation of the expected score for each arm, we approximate the average of the increase or decrease of the F1 value. Specifically, we calculate the difference between the F1 value of the classifier based on the labeled dataset constructed by a certain arm at time t and the F1 value of the classifier based on the dataset constructed at time $t-1$. Using the difference between the F1 values, we consider that a highly effective annotation action reflects the intuition that it will continue to improve the performance of the classifier. Additionally, when the improvement range of the arm that has been selected decreases or deteriorates as the annotation progresses, this may suggest that the nature of coders or labeled datasets is changing, and this leads to an opportunity to select another arm that matches the current annotation environment and improves the annotation. When reflecting the change of the nature of coders from the viewpoint of the time series, it can be helpful to use only the results in a certain section from the present to the past n times or gradually reduce the influence of past results when calculating the expected score of each arm. If only the results of a certain

interval from the present are used, the expected score for the arm that was not selected even once during that period cannot be calculated. Therefore, we adopt a strategy to reduce the influence of past results and use a decay function.

The formula for the expected score for each arm incorporating these elements is as follows:

$$Score\_Arm_i = \frac{\sum_{t=1}^{T} \Delta F_i(t) \times e^{-\alpha(T-t)}}{\sum_{t=1}^{T} c_i(t) \times e^{-\alpha(T-t)}} \tag{1}$$

$$\Delta F_i(t) = \begin{cases} F_t - F_{t-1} & (if\ Arm_i\ is\ operated\ at\ time\ t) \\ 0 & (otherwise) \end{cases} \tag{2}$$

$$c_i(t) = \begin{cases} 1 & (if\ Arm_i\ is\ operated\ at\ time\ t) \\ 0 & (otherwise)' \end{cases}$$

where T is the total number of annotation actions and F_t is the F1 value of the classifier constructed using the labeled dataset in time t. If all labels in the dataset are negative or the precision at time t cannot be computed, we set $F_t = 0$. We use $e^{-\alpha(T-t)}$ as the decay function, where α is the weighting parameter used for adjusting the damping speed. The weight for the R-th past result from the present is $e^{-\alpha R}$. If we set α so that this weight becomes W, then

$$\alpha = -\frac{1}{R} \log W.$$

For example, if the weight for the past 100th result from the present is 0.1, then α is $-1/100 \times \log 0.1$.

3.2 Algorithm

In this section, we describe a method for sequentially selecting a coder and text based on the ε-greedy algorithm. Table 1 contains a summary of the notation used in our method. Note that our method needs to set C, D, F^{th}, N, R, W and ε in advance. Additionally, D_1, D_2, and A are initially set to the empty set ∅. In our method, if Arm 1 is operated, the algorithm removes the selected text from D and adds the text with the label annotated by coder 1 to D_1. If Arm 2 is operated, the algorithm removes the selected text from D and adds the text with the label annotated by coder 2 to D_2. If Arm 3 is operated, the algorithm removes the selected text from D_1 and adds that text with the label by the adjudication to A. If Arm 4 is operated, the algorithm removes the selected text from D_2 and adds that text with the label by the adjudication to A.

Our method consists of three steps. We describe the details of each step below.

(Step 1) Initial setting for the classifier

As shown in Eq. (1), the expected score of each arm is calculated based on the F1 value of the classifier constructed from the labeled dataset. Therefore, before executing the ε-greedy algorithm, it is necessary to construct a scenario in which the F1 value can be calculated from the labeled dataset. Additionally, a classifier with an extremely low F1 value can be said to have poor dataset quality, which may adversely affect the calculation of the expected score. Furthermore, from the findings in [11],

Table 1. Notation.

	Description
D	Ordered set of texts to be annotated.
D_1	Set of pairs of text and its annotation by coder 1.
D_2	Set of pairs of text and its annotation by coder 2.
A	Set of pairs of text and its annotation by the two coders' adjudication.
C	Number of annotations that can be executed with the given budget.
F^{th}	Threshold for F1 value of classifier constructed by the labeled dataset.
N	Parameter for selecting each arm N / 4 times to evaluate evenly.
ε	Parameter for determining "exploitation" or "exploration".

the F1 value tends to increase as the amount of data increases for the initial annotation cost. Therefore, in step 1, the following process is executed.

First, Arm 1 and Arm 2 are selected alternately when D is not empty, and Arm 3 and Arm 4 are selected alternately when D is empty. After the arm is operated, C is decremented by 1. If there is at least one positive example and negative example in the labeled dataset and the F1 value of the classifier constructed using the labeled dataset is greater than or equal to F^{th}, the algorithm proceeds through step 2 and sets F_0 to the F1 value of the classifier constructed using the labeled dataset constructed. Note that this process is not regarded as an annotation action by the arm. If C is 0, our method is terminated.

(Step 2) Initial setting of the expected score for each arm

In this step, the initial score for $Score\_Arm_i$ $(1 \leq i \leq 4)$ is obtained. Each arm is essentially selected in the order Arm 1, Arm 2, Arm 3, and Arm 4. If there is no text to execute for the selected arm, the operation on the arm is skipped and N is decremented by 1, and then the next arm is selected (this process is repeated until a viable arm is selected). After the arm operation is complete, the F1 value of the classifier constructed using the labeled dataset is assigned to F_t, N is decremented by 1, and C is decremented by 1. If N is 0, the algorithm proceeds to step 3. If C is 0, our method is terminated.

(Step 3) Arm selection based on the ε-greedy algorithm

This step selects the arm based on the ε-greedy algorithm. First, r is randomly selected from [0, 1). If r is less than ε, the algorithm switches to "exploration," and if r is greater than or equal to ε, the algorithm switches to "exploitation." In "exploration" and "exploitation," the following processes are conducted.

- **Exploration**: The algorithm randomly selects one arm from the four types of arms with equal probability. If no text is available for annotation by the selected arm, the other arm is randomly selected again. This process is performed repeatedly until a viable arm is selected.
- **Exploitation**: The algorithm calculates $Score\_Arm_i$ $(1 \leq i \leq 4)$ using Eq. (1), and then selects the arm with the highest score for $Score\_Arm_i$. If no text is available for annotation by the selected arm, another arm with the next highest score for $Score\_Arm_i$ is selected (this process is performed repeatedly until a viable arm is selected).

After "exploration" or "exploitation," F_t is calculated. Then, C is decremented by 1, and the end condition in step 3 (whether C is 0) is confirmed.

4 Experiment

4.1 Experimental Settings

Our method can be applied to any binary classification task, but for our experiments, we chose to use the annotation task in [11]. This task was set originally to support social science research on the role of human values in the nuclear power debate, and was part of a multistep process that was ultimately intended to provide automated assistance for topically focused investigations of the role of human values in public debates. In this study, 120 Japanese newspaper editorials from the Mainichi Shimbun newspaper that substantively addressed the nuclear power debate in Japan following the Fukushima Daiichi disaster were divided into six subsets of 20, and each sentence in each document was annotated independently by two coders (both of whom were native Japanese speakers) for its relevance to eight predefined human value categories. Table 2 shows the definitions of the human value categories. For example, if a sentence described a judgement or evaluation based on results that included future prospects or macro/long-term perspectives, the sentence was considered to be positive for the human value "Consequence." Note that this annotation process allows multiple human value categories to be assigned to a sentence because more than one human value may be expressed in a sentence. The coders then decided the adjudicated annotation of a batch of 20 documents by discussion in one sitting. They repeated these processes six times because there were 120 documents. The resulting dataset that was created for each human value category by this annotation process contained three types of annotation for each sentence: coder A's annotation labels, coder B's annotation labels, and the adjudicated annotation labels that were ultimately arrived at through discussion between the coders.

In our experiment, annotated sentences in rounds 2 to 5 were used as training data and annotated sentences in round 6 were used as evaluation data. Sentences in round 1 were not used to minimize annotator learning effects. Additionally, six types of categories with more than 50 positive examples in the adjudicated training data for eight types of human value categories were chosen. Table 3 shows the details of the dataset for each human value. Each training data cell shows the number of sentences annotated by coder A/coder B/adjudication. Each evaluation data cell shows the number of sentences annotated as a result of the adjudication; that is, we regarded the adjudication in the evaluation data as the gold standard. The sizes of training and evaluation data were 2,188 and 570, respectively. In our method, 2,188 ordered sentences were set in D, described in Table 1, and 540 sentences were reserved for the calculation of the actual F1 values of the classifier constructed from the labeled dataset. Additionally, the annotation cost for annotating 2,188 sentences by adjudication was 4,376. To show the full range of results, we set C, described in Table 1, to 4,376.

Because our task was order dependent, we repeated the entire process 1,000 times, with 1,000 random shuffles of the sentences in the training data document-by-

Table 2. Definitions of the human value categories.

Human Value	Definition
Consequence	Values on judgement or evaluation based on results including future prospects (e.g. outcomes, objectives) or macro/long-term perspectives.
Safety	Values of safety and security; the state of being free from danger, injury, threat or fear; measures to prevent accidents and hazards.
Human Welfare	Values on fulfilling benefits common to human beings and related to society as a whole; Clear benefits to the public.
Intention	Values on emotion or feelings including impression, attitude, empathy, prudence, and sincerity; The quality of being honest and integrity; adherence to moral principles.
Social Order	Values on social structure, including rules, norms, common sense and expectations as well as social responsibility; Institutional, legal, and political decisions involving governments and states.
Wealth	Values on pursuing any economic goals, such as money, material possessions, resources, and profit including business activities.
Freedom	Value of individual freedom and choices; the state of being unconstrained; freedom from interference or influence by others;
Personal Welfare	Values on personal needs, growth, and self-actualization.

Table 3. Distribution of the dataset for each human value category.

	Training Data		Evaluation Data	
	Positive	Negative	Positive	Negative
Consequence	667/758/959	1521/1430/1229	270	270
Safety	586/663/719	1602/1525/1469	249	291
Human Welfare	196/196/224	1992/1992/1964	73	467
Intention	167/152/205	2021/2036/1983	66	474
Social Order	1445/1473/1570	743/715/618	431	109
Wealth	263/251/289	1925/1937/1899	118	422

document. Each time a sentence was annotated, we retrained a support vector machine (SVM), implemented in TinySVM[2] using a linear kernel. We used Juman version 7.01[3] to tokenize and perform morphological analysis for Japanese, and we used all the resulting words as features for the classifier after removing period and comma characters. We used whether the word appeared or not as each feature's weight.

We plotted learning curves for the evaluation data by placing the annotation cost on the horizontal axis and the F1 value on the vertical axis. Note that the horizontal axis represents the number of annotations, not the number of sentences, because one sentence may have two annotations. The learning curves were for the following:

- **Adjudicated (baseline)**: we used the adjudicated data for training, and counted this as two annotations in each iteration.
- **Alternating (baseline)**: we alternated between using coder A's and coder B's individual annotations for training until all sentences were annotated, and then replaced single annotations with adjudicated annotations in the same order.

[2] http://chasen.org/~taku/software/TinySVM/

[3] http://nlp.ist.i.kyoto-u.ac.jp/EN/index.php?JUMAN

- **Selection**: we sequentially selected the next sentence and coder using the ε-greedy algorithm.

We plotted the above learning curves for each human value category in Table 3.[4]

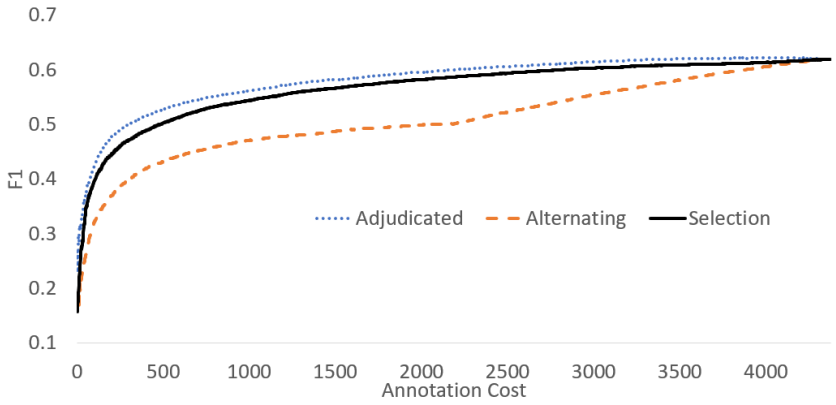
We determined the parameters F^{th}, N, R, W, and ε used in our method using the following approach. We used the datasets for Consequence, Safety, and Human Welfare as tuning data, and repeatedly performed the following: we observed the transition of F1 values in the three types of dataset and manually tuned each parameter based on the results. We fixed N at 40 because step 2 played a part in obtaining the initial value for each arm used in step 3, and it was not necessary to incur a large cost in step 2 because the score of each arm was updated in step 3. Additionally, we fixed ε at 0.1, which has been commonly used in various tasks that used the ε-greedy algorithm. Finally, we manually determined the parameters with good F1 value transitions for the three types of dataset. The parameters we determined were $(F^{th}, N, R, W, \varepsilon)$ = (0.3, 40, 200, 0.1, 0.1). We applied this combination of parameters to the other three types of dataset (Intention, Social Order, and Wealth).

4.2 Experimental Results

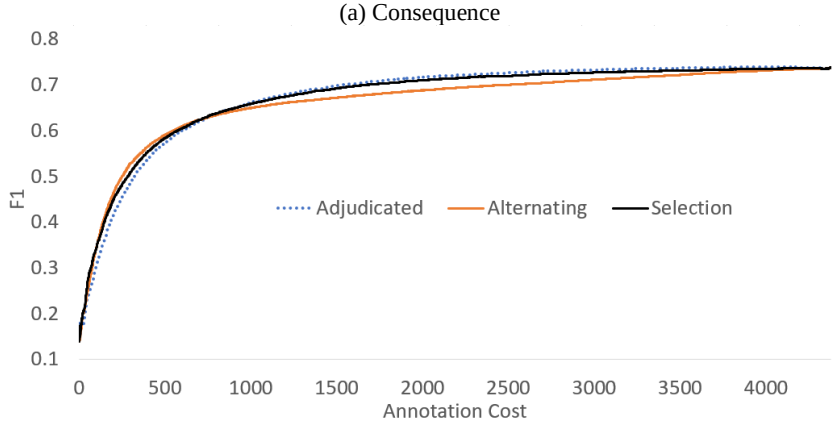
Fig. 1 shows how F1 improved based on the macro average over 1,000 random simulations as the number of annotations increased for each human value. First, we observe the results when the parameters described in Section 4.1 were tuned on the same categories (Consequence, Safety, Human Welfare). For Consequence (Fig. 1(a)), the Selection curve shows F1 values close to those of the Adjudicated curve, which achieved higher values than the Alternating curve for the early annotation cost. For Safety (Fig. 1(b)) and Human Welfare (Fig. 1(c)), the Selection curve shows F1 values close to those of the Alternating curve, which achieved higher values than the Adjudicated curve for the early annotation cost, and then later shows almost the same F1 values as the Adjudicated curve, which achieved higher values than the Alternating curve, because the annotation costs where the transitions of the F1 value occurred when using the Adjudicated and Alternating curves intersected.

Next, we investigate the results when the optimum parameters determined using the above categories were applied to the other categories (Intention, Social Order, Wealth). For Intention (Fig. 1(d)), the Selection curve shows F1 values close to those of the Alternating curve, which achieved higher values than the Adjudicated curve for the early annotation cost. For Social Order (Fig. 1(e)), the Selection curve shows F1 values close to those of the Adjudicated curve, which achieve higher values than the Alternating curve for the early annotation cost. For Wealth (Fig. 1(f)), the Selection curve shows almost the same F1 values as the Alternating curve, which achieved higher values than the Adjudicated curve for the early annotation cost (annotation cost interval 1–600), and then shows F1 values between the Adjudicated and Alternating curves. These results indicate that our method generally yielded good results that

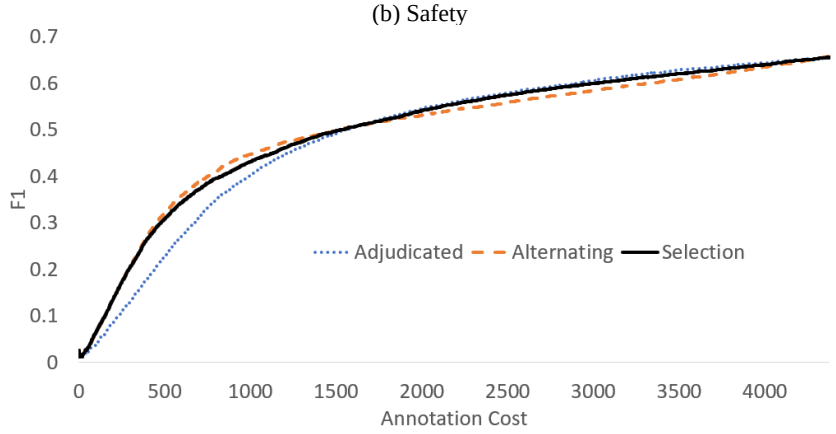
[4] Note the difference in annotation strategy between constructing annotated data in [11] and applying the multi armed bandit (MAB)-problem method: in [11], coders assigned several labels to each sentence in one sitting; however for the MAB-problem method, one label is assigned or not for each sentence.



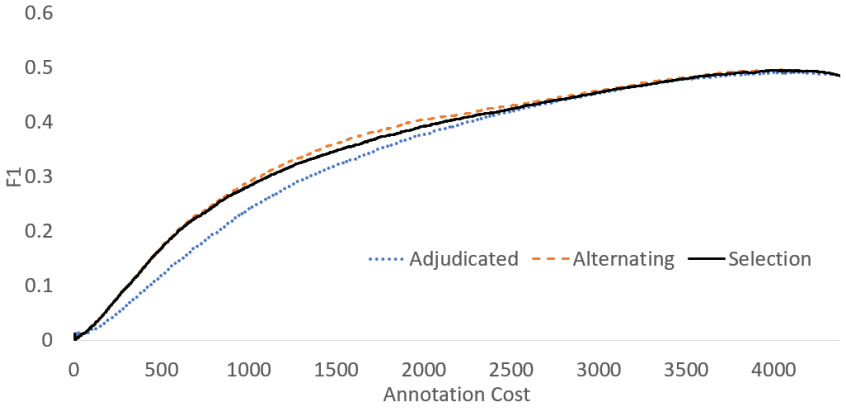
(a) Consequence



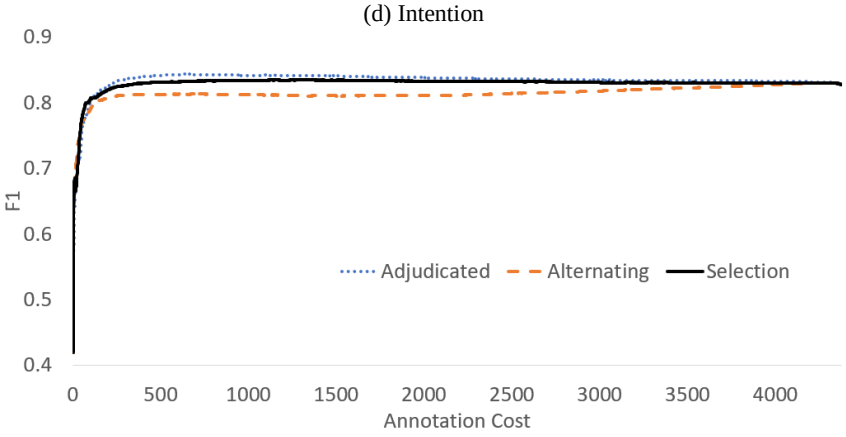
(b) Safety



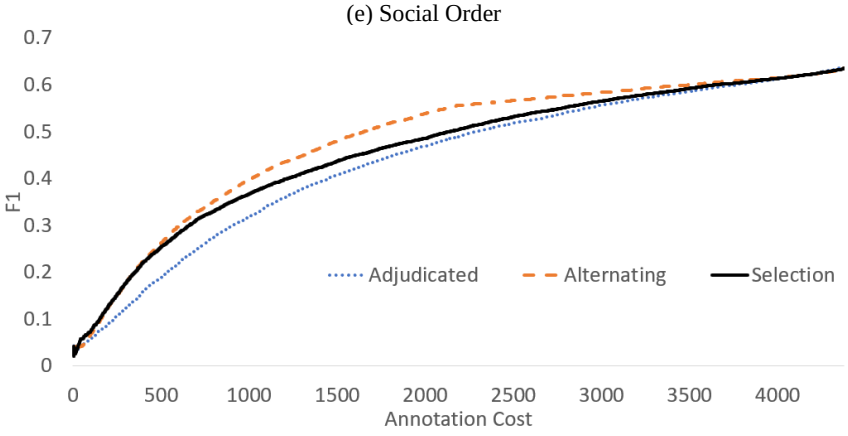
(c) Human Welfare



(d) Intention



(e) Social Order



(f) Wealth

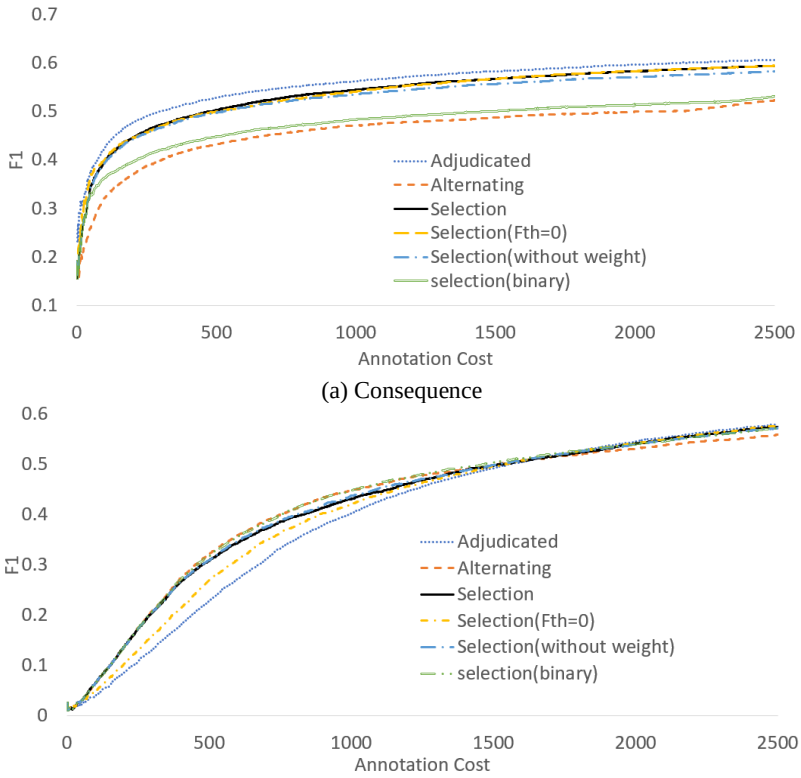
Fig. 1. Learning curves for each human value.

were between the Adjudicated and Alternating curves. Additionally, to further improve the learning curves of Selection for categories such as Wealth, we would need to automatically estimate optimum parameters according to task and work scenario.

4.3 Discussion

In this section, we verify the effectiveness of F^{th}, $e^{-\alpha(T-t)}$, and difference between the F1 values ($\Delta F_i(t)$) in Eq. (2) in Selection to investigate how these elements added to or changed the conventional ε-greedy algorithm's effect on the F1 value's curve. For these verifications, we used the datasets of Consequence and Human Welfare, which were used for parameter tuning. For F^{th}, we set $F^{th} = 0$ to verify whether it is better to move to step 2 with a value above a certain level for the F1 value of the classifier constructed from the training data created in step 1. Note that all parameters other than F^{th} used the values described in Section 4.1. For $e^{-\alpha(T-t)}$, we compared the cases with and without this weight in Eq. (1). Note that all parameters other than R and W used the values described in Section 4.1. For $\Delta F_i(t)$, we compared the case where the score follows the difference between F1 values with the case where the score follows the Bernoulli distribution used in the conventional bandit algorithm. In this verification, we set $\Delta F_i(t)$ to 1 if the F1 value improved from the previous classifier, and 0 otherwise. For this case, we used the parameter values described in Section 4.1.

Fig. 2 shows the transitions of the F1 values for the above cases. Selection (Fth = 0) shows the curve when F^{th} is set to 0 in Selection. Selection (without weight) shows the curve when $e^{-\alpha(T-t)}$ is not used in Eq. (1) in Selection. Selection (binary) shows the curve when binary evaluation is used instead of the difference between the F1 values ($\Delta F_i(t)$) in Eq. (2) in Selection. First, we compare the results for the settings of F^{th}. For Consequence (Fig. 2(a)), the transitions of the F1 values when $F^{th} = 0$ and $F^{th} = 0.3$ were almost the same. However, for Human Welfare (Fig. 2(b)), the results for $F^{th} = 0.3$ demonstrated higher performance than the results for $F^{th} = 0$. From these results, we conclude that to construct a high-performance classifier consistently in Selection, we need to construct a classifier that has a certain F1 value before moving to step 2 in Selection. Next, we compare the results with and without $e^{-\alpha(T-t)}$. In Human Welfare (Fig. 2 (b)), the effect of using weights was not confirmed; however, higher F1 values were confirmed when the weight was used for Consequence (Fig. 2(a)). These results suggest that the probabilities of the improved performance of some arms may differ between past and current work, depending on the task. Therefore, we can possibly calculate the expected score of each arm more precisely by reducing the influence of past work. Finally, we compare the results when using the difference between F1 values and when using the binary values 1 (F1 value improved) or 0 (otherwise) in Eq. (2). For Human Welfare (Fig. 2(b)), the F1 value was relatively good when the binary values were used; however, the F1 values were very low, close to those of Alternating, in Consequence (Fig. 2(a)). These results suggest that the binary indicator used in the conventional bandit algorithms to indicate whether the F1 value had improved did not accurately measure the expected score of the arm for some tasks, and it was better to use the elaborated score by evaluating how much the F1 value had improved or deteriorated.



(a) Consequence

(b) Human Welfare

Fig. 2. Learning curves for changes in various settings in the annotation cost interval 1–2,500.

5 Conclusion

We have demonstrated that a high-performance classifier can be consistently and affordably constructed for a range of annotation tasks using a multi-armed bandit algorithm to select the coder(s) when creating a labeled training set. The key idea behind our method is to automatically select the coder that the annotation history would lead us to expect would be the most likely to increase the classifier's effectiveness, measured in our experiments using F1. Over six annotation tasks, we showed that this approach tends to show the better of two reasonable baselines. In the future, we plan to apply this technique to other digital library applications, notably including topic classification, we plan to integrate active learning approaches so that we select not just the best choice of coder but also the best choice of document to be coded, we plan to explore approaches to further improve efficiency using asynchronous batch updates, and we plan to explore approaches to efficiently tune model parameters to specific application settings.

Acknowledgements. This work was supported by JSPS KAKENHI Grant Number JP18H03495.

References

1. Artstein, R., Poesio, M.: Inter-coder agreement for computational linguistics. Computational Linguistics 34(4), 555–596 (2008).
2. Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. Machine Learning 47(2-3), 235–256 (2002).
3. Bennett E.M., Alpert R., Goldstein A.C.: Communications through limited response questioning. Public Opinion Quarterly 18(3), 303–308 (1954).
4. Cai, W., Zhang, Y., Zhou, J.: Maximizing expected model change for active learning in regression. In: Proc. of ICDM, pp. 51–60 (2013).
5. Carletta, J.: Assessing agreement on classification tasks: the kappa statistic. Computational Linguistics 22(2), 249–254 (1996).
6. Cohen, J.: Weighted kappa: nominal scale agreement with provision for scaled disagreement or partial credit. Psychological Bulletin 70(4), 213–220 (1968).
7. Culotta, A., McCallum, A.: Reducing labeling effort for structured prediction tasks. In: Proc. of AAAI, pp. 746–751 (2005).
8. Fort, K., François, C., Galibert, O., Ghribi, M.: Analyzing the impact of prevalence on the evaluation of a manual annotation campaign. In: Proc. of LREC, pp. 1474–1480 (2012).
9. Garivier, A., Moulines, E.: On upper-confidence bound policies for switching bandit problems. In: Proc. of the International Conference on Algorithmic Learning Theory, pp. 174–188 (2011).
10. Howe, J.: Crowdsourcing: why the power of the crowd is driving the future of business. Crown Publishing Group (2008).
11. Ishita, E., Fukuda, S., Oga, T., Tomiura, Y., Oard, D.W., Fleischmann, K.R.: Cost-effective learning for classifying human values. In: Proc. of iConference (2020).
12. Kuriyama, K., Kando, N., Nozue, T., Eguchi, K.: Pooling for a large-scale test collection: an analysis of the search results from the first NTCIR workshop. Information Retrieval 5(1), 41–59 (2002).
13. Nguyen, A.T., Wallace, B.C., Lease, M.: Combining crowd and expert labels using decision theoretic active learning. In: Proc. of HCOMP, pp. 120–129 (2015).
14. Raj, V. and Kalyani, S.: Taming nonstationary bandits: A bayesian approach. arXiv preprint arXiv:1707.09727 (2017).
15. Scott, W.: Reliability of content analysis: The case of nominal scale coding. Public Opinion Quarterly 19, 321–325 (1955).
16. Thompson, W.R.: On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. Biometrika 25(3-4), 285–294 (1933).
17. Voorhees E.M., Harman D.K.: TREC: experiment and evaluation in information retrieval. The MIT Press (2005).
18. Welinder, P., Branson, S., Belongie, S., Perona, P.: The multidimensional wisdom of crowds. In: Proc. of NIPS, pp. 2424–2432 (2010).
19. Zhang, Y., Cui, L., Huang, J., Miao, C.: CrowdMerge: achieving optimal crowdsourcing quality management by sequent merger. In Proc. of ICCSE, pp. 1–8 (2018).