Targeted Feedback Collection Applied to Multi-Criteria Source Selection

Julio César Cortés Ríos(✉), Norman W. Paton, Alvaro A.A. Fernandes, Edward Abel, and John A. Keane

School of Computer Science, University of Manchester,
Oxford Road, Manchester M13 9PL, UK
juliocesar.cortesrios@manchester.ac.uk

Abstract. A multi-criteria source selection (MCSS) scenario identifies, from a set of candidate data sources, the subset that best meets a user's needs. These needs are expressed using several criteria, which are used to evaluate the candidate data sources. A MCSS problem can be solved using multi-dimensional optimisation techniques that trade-off the different objectives. Sometimes we may have uncertain knowledge regarding how well the candidate data sources meet the criteria. In order to overcome this uncertainty, we may rely on end users or crowds to annotate the data items produced by the sources in relation to the selection criteria. In this paper, we introduce an approach called Targeted Feedback Collection (TFC), which aims to identify those data items on which feedback should be collected, thereby providing evidence on how the sources satisfy the required criteria. TFC targets feedback by considering the confidence intervals around the estimated criteria values. The TFC strategy has been evaluated, with promising results, against other approaches to feedback collection, including active learning, using real-world data sets.

Keywords: Data integration · Source selection · Feedback collection · Pay-as-you-go · Multi-objective optimisation

1 Introduction

The number of available data sources is increasing at an unprecedented rate [8]. Open data initiatives and other technological advances, like publishing to the web of data or automatically extracting data from tables and web forms, are making the source selection problem a critical topic. In this context, it is crucial to select those data sources that satisfy user requirements on the basis of well-founded decisions.

Regarding the properties that the data sources must exhibit, there have been studies of the source selection problem considering specific criteria, such as accuracy, cost and freshness [5,18]. In this paper, we deploy a multi-criteria approach that can be applied to diverse criteria in order to accommodate a wider variety of user requirements and preferences, while considering the trade-off between

M. Kirikova et al. (Eds.): ADBIS 2017, LNCS 10509, pp. 136–150, 2017.
DOI: 10.1007/978-3-319-66917-5_10

the required criteria. In this approach, from a collection of sources, S, the problem is to identify a subset of the sources S' from which R data items can be obtained that reflect the user's preferences. These preferences are represented by a collection of weighted criteria; for example, the criteria could be of the form *accuracy:0.4, freshness:0.3, relevance:0.3*, indicating that *freshness* and *relevance* are of equal importance to the user, and that *accuracy* is more important still.

To solve the multi-dimensional source selection problem, where each dimension represents a different criterion relating to the data sources, a multi-dimensional optimisation technique is used to provide a solution that takes into account the user's preferences (represented as weights) in relation to the criteria. This Multi-Criteria Source Selection problem (MCSS) has been addressed before (e.g. [16,17]). This paper addresses the MCSS problem using an approach where the objective is to retrieve an optimal number of items from each supplier given the weighted criteria [21] that model a user's requirements.

To inform the source selection process, where criteria estimates are likely to be unreliable, we need to annotate the candidate data sources to obtain their criteria values; this is an essential step, as we need to know how each source scores for each criterion. Given that there may be many sources and criteria, the source annotation process can become expensive. In this paper, we focus on pay-as-you-go approaches, and collect feedback in the form of true and false positive annotations on data items, that indicate whether or not these data items satisfy a specific criterion. Such feedback could come from end users or crowd workers, and has been obtained in previous works [1,2,7,19].

Having an efficient way to collect the feedback required to improve our knowledge of the data sources is important, as there are costs involved. Hence, we need to carefully identify the data items on which to ask for feedback in order to maximise the effect of the new evidence collected, and to minimise the amount of feedback we need to collect. Some recent work has focused on targeting feedback, especially in *crowdsourcing* (e.g. [4,9,14]); here we complement such work by providing an approach to feedback collection for multi-criteria source selection.

In this paper, we build upon the same statistical foundation as [19], which also targets feedback in a way that takes into account the margins of error in criteria estimates. However, in [19] the approach targeted feedback in a setting where there was a trade-off between two fixed criteria (precision and recall), which was explored using single-dimensional, constrained optimisation. In contrast, here there may be arbitrary numbers of criteria, these criteria are weighted, and the search for solutions involves multi-dimensional, constrained optimisation. So, in comparison with [19], the problem solved in this paper is harder in the sense that a solution must satisfy a variable number of weighted criteria of different types (and not only in terms of data quality).

The following contributions are reported in this paper: (i) a strategy for targeted feedback collection for use with MCSS in which there are arbitrary numbers of weighted criteria; (ii) an algorithm that implements the strategy, using the confidence intervals around the criteria estimates to identify those

sources that require more feedback to improve the results of MCSS; and (iii) an experimental assessment of our approach using real-world data to show that TFC can consistently and considerably reduce the amount of feedback required to achieve high-quality solutions.

2 Problem Description

MCSS is a complex problem when the number of criteria is large and the user can declare preferences over these criteria. Concretely, the MCSS problem can be defined as: given a set of candidate sources $S = \{s_1, \ldots, s_m\}$, a set of user criteria $C = \{c_1, \ldots, c_n\}$ with weights $W = \{w_1, \ldots, w_n\}$, and a target number of data items R, identify a subset S' of S that satisfies the requirements expressed by the user while maximising the combined criteria. The solution is presented as a collection $\mathbf{X}$ with m elements, indicating how many data items each source in S contributes to the solution. Sources in $S \setminus S'$ do not contribute.

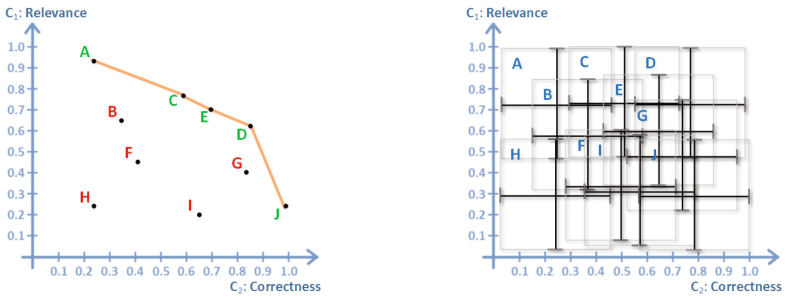

Fig. 1. Multi-criteria data source selection without (a) and with (b) uncertainty.

Consider the example presented in Fig. 1(a), in which there are 10 data sources $S = \{A, B, C, D, E, F, G, H, I, J\}$, and 2 data property criteria to balance: relevance (c_1) and correctness (c_2) with the following weights $W = \{w_1 = 0.5, w_2 = 0.5\}$. The user requires a particular number of data items (R) from a subset of sources in S that maximise both criteria and reflect the weights W (in this case, the user considers the two criteria to be of identical importance).

This problem can be solved by using linear programming or other multi-dimensional optimisation techniques. In our case, we are considering an additional factor, the presence of uncertainty in the source criteria estimates. This uncertainty is caused by incomplete evidence, and can be reduced by annotating data items produced by each source to determine if they satisfy the conditions for each criterion. We developed the TFC strategy to identify the data items that reduce this uncertainty to support better solutions for the MCSS problem.

To solve the MCSS problem we can apply an optimisation technique, such as that presented in Subsect. 3.3, and obtain a solution $\mathbf{X}$ which is a vector containing the values of the decision variables x after the optimisation for all the sources. This solution indicates how many data items each data source from S contributes. In Fig. 1(a) we draw a line to highlight the sources in $S' = \{A, C, D, E, J\}$, that in our example contribute data items to the solution.

We now consider the case where we have uncertain knowledge about the criteria values. This brings an even more complex problem. In Fig. 1(b), instead of dots representing the data sources in the multi-criteria space, we have bi-dimensional intervals representing the uncertainty around each source's criteria values. The real criterion value may be expected to lie within this area, but, in the absence of more evidence, we do not know exactly where. Therefore, now the question is: how can we cost-effectively select data items to collect feedback on, in order to reduce the uncertainty in a way that benefits the optimisation technique in solving the MCSS problem? We propose TFC, an approach that minimises the number of data items on which we need to collect feedback and determines the point beyond which the collection of more feedback can be expected to have no effect on which sources were selected.

3 Technical Background

3.1 Data Criteria

A data criterion is a metric that can be applied to a data set to evaluate how the data set fares on that criterion. There are many different criteria that can be applied to the data source selection problem. For instance, in [15,17] accuracy (degree of correctness) and freshness (how recent is the information) were used.

In this paper, we evaluate the estimated value of a data criterion $\hat{c}$ as the ratio between the elements satisfying the notion for a given metric (and for which feedback has been collected) or true positives, tp, and all the elements that have been annotated (which is the sum of the true and false positives, fp). For example, to evaluate the relevance of a source we divide the number of relevant data items over the total number of data items labelled for that source. We use the following formula to compute the ratio of expected data items over all the data items on which feedback has been collected for a source s:

$$\hat{c}_s = \frac{|tp_s|}{|tp_s| + |fp_s|} \tag{1}$$

3.2 Confidence Interval, Overlap and Sample Size

Our strategy is based on classifying the candidate sources into those that are suitable (given a collection of data criteria) to include in a solution, and those sources that are not. This classification is done by analysing the overlapping of the confidence intervals around the criteria value estimates for each source. A *confidence interval* is the range formed by flanking the estimated value (based on the

available evidence or feedback) with a margin of error for a required confidence level and represents the space in which the true value is expected to be contained. This confidence interval is associated with a given *confidence level*, which is the percentage of all possible samples that can be expected to include the real value. Following this definition, the larger the number of data items we have labelled for a source, the greater the confidence in the estimated values, and hence, the smaller the confidence intervals around these estimates. We use the following formulae to compute the confidence intervals for a source s [3] (assuming the data is normally distributed as the real distribution is unknown):

$$se_s = \sqrt{\frac{\hat{c}_s \cdot (1 - \hat{c}_s)}{L_s}} \tag{2}$$

$$fpc_s = \sqrt{\frac{T_s - L_s}{T_s - 1}} \tag{3}$$

$$e_s = z_{cL} \cdot se_s \cdot fpc_s \tag{4}$$

To compute the upper and lower bounds of the confidence interval we have:

$$upCI = \min(\hat{c}_s + e_s, 1.0) \tag{5}$$

$$lowCI = \max(\hat{c}_s - e_s, 0.0) \tag{6}$$

where s is a source in the set of candidate data sources S, se_s is the *standard error*, fpc_s is the *finite population correction factor* used to accommodate data sources of any size assuming that they have a finite number of data items, L_s is the number of feedback instances collected for s, T_s is the total number of data items produced by s, $\hat{c}_s$ is the estimated data criterion, and $lowCI$ and $upCI$ are the lower and upper bounds of the confidence interval, respectively. The result is the *margin of error* e_s around our estimate, e.g. $\hat{c}_s \pm e_s$, for a given confidence level cL and its corresponding z–score z_{cL}.

An important part of our strategy relies on the confidence intervals surrounding the criteria estimates for each source, and how these confidence intervals overlap. The approach [10] is to determine not only if two confidence intervals overlap, but also the amount of overlapping. Analysis of this overlapping helps in determining whether two intervals are significantly different or else can be considered as equivalent. In our approach we only consider intervals which are *significantly overlapping*, i.e. if the values are significantly different (thereby providing strong evidence). The estimated values are significantly different if:

$$\hat{c}_{s_1} - \hat{c}_{s_2} > z_{cL} \cdot \sqrt{se_{s_1}^2 + se_{s_2}^2}, \tag{7}$$

and there is no overlap between confidence intervals if:

$$\hat{c}_{s_1} - \hat{c}_{s_2} > z_{cL} \cdot (se_{s_1} + se_{s_2}). \tag{8}$$

The TFC strategy uses the notion of *sampling size* to compute the number of data items required to obtain a representative sample of the entire population

given a confidence level and a margin of error. Feedback on the sample elements is then obtained to establish an initial understanding of the underlying data quality. The sample size sS for a finite population T [6,12] is also used to estimate the number of elements required during each feedback collection episode:

$$sS_T = \frac{ssInf}{1 + \frac{ssInf-1}{|T|}} \tag{9}$$

which is based on the formula for the sample size of a very large (infinite) population for a given margin of error e and a desired confidence level cL:

$$ssInf = \frac{z_{cL}^2 \cdot (\hat{c}_s \cdot (1 - \hat{c}_s))}{e^2} \tag{10}$$

3.3 Multi-criteria Optimisation

Regarding the multi-dimensional or multi-criteria optimisation techniques, we consider that the data criteria are evaluated using linear functions and therefore linear-programming techniques can be used to find the optimal solution that balances all the criteria and user preferences (represented as weights). To solve the optimisation problem, we have selected *Min-sum*, a Multi-objective linear programming (MOLP) algorithm, whose objective is to maximise the overall weighted utility of the solution considering the minimum deviation λ from the objectives, and the trade-off between all the objectives. This can be represented as a linear programming problem with a collection of objective functions $\mathbf{Z}$ and their associated constraints, and the general goal of maximising the overall weighted utility of the solution. The solution is represented as the vector $\mathbf{X}$ containing the values of all the decision variables x after the optimisation.

First we need to obtain the ideal Z_k^* and negative ideal Z_k^{**} solutions (best and worst possible solutions respectively) for each criterion k by using single objective optimisation. These solutions are found by optimising each criterion with respect to the following single objective function Z:

$$Z_k = \frac{\sum_{i=1}^{m} x_i \cdot \hat{c}_{k_{s_i}}}{R} \quad k = 1, 2, \ldots, n, \tag{11}$$

where m is the number of candidate sources available, n is the number of user criteria, x_i is the number of data items used from source s_i, $\hat{c}_{k_{s_i}}$ is the value of the criterion k for source s_i, and R is the number of data items requested.

The objective function in Eq. 11 is solved as both maximisation and minimisation objective functions for each criterion k. These functions are constrained as follows. The number of data items chosen from each source in S, x_i, cannot exceed the maximum number of data items $|s|$ produced by that source:

$$x_i \leq |s_i| \quad i = 1, 2, \ldots, m \tag{12}$$

The total number of data items chosen must equal the amount requested:

$$\sum_{i=1}^{m} x_i = R \tag{13}$$

And the minimum value for the decision variables x is 0 (non-negativity):

$$x_i \geq 0 \quad i = 1, 2, \ldots, m \tag{14}$$

The ideal and negative ideal solutions for each criterion k are then computed to obtain the range of possible values. These solutions, along with the constraints from Eqs. 12–14, and the user preference weights w are used to find a solution that minimises the sum of the criteria deviations. Each per-criterion deviation measures the compromise in a solution with respect to the corresponding ideal value given the user weights.

The weighted deviation for each criterion D_k is computed by comparing how far the current solution is from the ideal solution, as follows:

$$D_k = \frac{w_k \cdot (Z_k^* - Z_k)}{Z_k^* - Z_k^{**}} \quad k = 1, 2, \ldots, n, \tag{15}$$

And finally, the optimisation model consists in minimising the sum of criteria deviations λ (measure of the overall deviation from the objective) by considering the constraints in Eqs. 12–15 as follows:

$$\min \quad \lambda, \lambda = D_1 + D_2 + \ldots + D_n, \tag{16}$$

4 Targeting Feedback Using Multi-dimensional Confidence Intervals

We now define the TFC strategy. Consider the MCSS example problem described in Sect. 2, where the goal is to select, from the available candidates, the data sources that provide the maximum combined relevance and correctness. For this goal, some budget was allocated to fund feedback on b data items. We assume no up-front knowledge of the relevance or correctness.

To solve this problem, we need some initial estimates about the values of the criteria for the candidate data sources as shown in Fig. 1(b). We obtain these by collecting feedback on a representative random sample of data items (Eq. 9). To compute the criteria estimates, we use Eq. 1, and calculate the associated margin of error with Eqs. 4, 5 and 6, to obtain the confidence intervals for each criterion for each data source as shown in Fig. 2(a).

Given these initial estimates, we address the goal of finding the combination of sources that maximises the desired criteria (relevance and correctness in the example) while considering the trade-off between them. We can formulate this objective as a Min-sum model using Eq. 16 with the criteria estimates as our coefficients and the number of data items from each source as the decision

variables. Min-sum then finds the combination of data items returned by each source that yields the maximum overall weighted utility oU for the optimisation goal (maximum combined relevance and correctness).

By applying Min-sum over the candidate sources, we find the subset of sources forming a non-dominated solution $Ss = \{A, C, D, G, J\}$. This preliminary solution is illustrated in Fig. 2(a). In the same figure, a different subset of sources $So = \{B, E, I\}$ is identified that are not part of Ss but that have confidence intervals for the optimisation criteria that overlap with sources in Ss (graphically, the areas defined by the confidence intervals of B, E and I overlap those defined by the the confidence intervals of the sources in the non-dominated solution, whereas F and H do not). This overlap is computed using Eqs. 7 and 8. It suggests that, in addition to the sources in the non-dominated solution, we need to collect more feedback on B, E, and I in order to decide whether they belong to the solution or not. Our strategy then collects more feedback on a new set, $S' = Ss \cup So = \{A, B, C, D, E, G, I, J\}$. The sources in S' benefit from additional feedback either by reducing the uncertainty as to whether or not they should contribute to the solution, or by refining their criteria estimates.

Having decided on the data sources we need to collect more feedback on, we determine how much feedback should be obtained. This is obtained with Eq. 9, which computes the sample size over a population which, in this case, are the unlabelled data items produced by all the sources in S'.

Having decided on the number of data items that we need to collect feedback on, we collect the feedback and use it to refine our criteria estimates; this is done by recalculating the estimates and margin of error for each criterion for each source. We follow this approach for the sources which are either part of a preliminary solution or are candidates to become part of the solution.

This refinement continues while we have enough budget b for additional feedback collection, there is still some overlap between the confidence intervals of sources contributing to the solution and those from non-contributing sources, and there are still unlabelled data items.

It is important to notice that in Fig. 2(a), some sources are not considered for further feedback collection (viz., F and H), since their confidence intervals do not overlap with those of any of the sources contributing to the solution, and therefore, they have no statistical possibility of being part of it, unless there is a need for more data items than those produced by sources in S'. By filtering out these outliers, TFC focuses on those sources that can be part of the solution.

The strategy leads to a state where the confidence intervals for sources in the solution do not overlap with the intervals of other sources, as shown in Fig. 2(b). The result is a solution with low error estimates and a set of data sources that were excluded from additional feedback collection as they have a low likelihood of being part of an improved solution.

An important feature of our strategy is that it can be applied to problems with multiple criteria, varied user preferences (weights) for each criterion, and over a large number of data sources of variable size and quality.

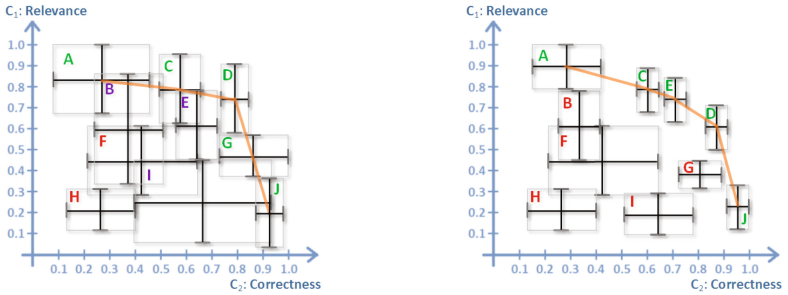

Fig. 2. Confidence intervals with overlapping (a) and without overlapping (b)

5 Algorithm

In this section, we describe our algorithm for the TFC strategy applied to the MCSS problem. The pseudo-code for the algorithm is given in Fig. 3.

The inputs for the algorithm are: S: the collection of sources from which we need to select a subset that together satisfy the user requirements considering the criteria and specific preferences; C: the collection of criteria modelled as described in Subsect. 3.1; U: the set of unlabelled data items produced by sources in S; W: the collection of criteria weights representing the user's preferences; b: the allocated budget for the total number of items on which feedback can be obtained; R: the total number of user requested data items; and for statistical estimations cL: the confidence level; and e: the initial margin of error. The output is a vector $\mathbf{X}$ with the number of data items each source contributes.

To solve the MCSS problem, based on the criteria estimates refined by our TFC approach, we first need to obtain the sample size for the number of data items that need to be annotated to achieve a statistically representative view of all candidate data items (line 3). We compute this sample size with Eq. 9 using the number of unlabelled data items produced by sources in S' to represent the sample population. The confidence level and margin of error determine the sample size of the data items on every source.

The function *collectFeedback* (line 4) represents the feedback collection process, which takes as arguments the set of sources considered for feedback S', the set of unlabelled data items U, and the number of additional data items on which feedback is required sS. This function randomly selects from U at most sS data items on which feedback needs to be collected. The remaining budget is updated accordingly depending on the number of data items identified (line 5).

The criteria values can be estimated for each candidate data source in S once the feedback is collected and assimilated. The function *estCriteriaValues* (line 6) uses the candidate data sources S, the sets of labelled and unlabelled data items L and U, the collection of criteria C, and a given confidence level and margin of error cL and e, to compute the collection of estimated criteria values $\hat{C}$ for

each data source. These estimates rely on the data items already labelled and are computed with Eq. 1, as described in Subsect. 3.1. The estimates obtained are used to build the confidence intervals (Eqs. 5 and 6) around each criterion estimate (and for each source), by computing the margin of error with Eq. 4. The confidence intervals are then analysed for overlapping one dimension at a time. We follow this approach to handle multiple dimensions without considering simultaneously for the statistical computations. An example of how the confidence intervals may look at this early stage of the process is shown in Fig. 1 from Sect. 2, where there is high overlapping between the confidence intervals and no clear candidate sources.

At this point, with initial estimated criteria values for all the candidate sources, the MCSS problem can be solved by applying an optimisation model as described in Subsect. 3.3 (Min-sum) to obtain a solution that maximises the overall weighted utility oU. The *solveMCSS* function (line 7) represents this step and requires the collection of candidate data sources S, the set of estimated criteria $\hat{C}$, the set of weights representing the user preferences W, and the total number of user requested data items R. The output from this optimisation is a vector $\mathbf{X}$ with the number of data items each candidate source contributes. The set S' is initialised before processing the candidate sources (line 8).

Input: set of sources S
Input: set of data property criteria C
Input: set of unlabelled data items U
Input: set of weights for the data property criteria W
Input: a budget size b
Input: a number of user requested data items R
Input: a confidence level cL
Input: a margin of error e
Output: vector $\mathbf{X}$ with no. of items contributed per source
1: $L \leftarrow \{\}$, $X \leftarrow \{\}$, $S' \leftarrow S$, $sS \leftarrow 1$
2: **while** $b > 0$ and $sS > 0$ and $S' <> \{\}$ **do**
3: $sS \leftarrow$ computeSampleSize(S', U, b, cL, e)
4: $L \leftarrow$ collectFeedback(S', U, sS)
5: $b \leftarrow b - sS$
6: $\hat{C} \leftarrow$ estCriteriaValues(S, L, U, C, cL, e)
7: $\mathbf{X} \leftarrow$ solveMCSS$(S, \hat{C}, W, R)$
8: $S' \leftarrow \{\}$
9: **for** $s \in S$ **do**
10: $oL \leftarrow false$
11: **for** $c \in C$ **do**
12: $oL \leftarrow$ isSignificantlyOverlapping$(s_c, \mathbf{X})$ or oL
13: **end for**
14: **if** $\mathbf{X}[s] > 0$ or oL **then**
15: $S' \leftarrow S' \cup s$
16: **end if**
17: **end for**
18: **end while**
19: **return** $\mathbf{X}$

Fig. 3. TFC algorithm

Having the confidence intervals for each criterion and data source, and the sources that contribute to a preliminary solution $\mathbf{X}$, we can analyse the overlap between these intervals. This analysis is performed in the *isSignificantly-Overlapping* function (line 12), which is called with the estimate for each criterion c in C applied to each data source s in S. The function also requires the solution for the MCSS problem $\mathbf{X}$ to determine which intervals from sources contributing to the solution significantly overlap with intervals from non-contributing sources. The overlapping analysis uses the concepts defined in Subsect. 3.2, in particular Eqs. 7 and 8 to determine if two intervals are significantly overlapping or not. As we evaluate this overlapping at source level (not criterion level), when at least one criterion is evaluated with significant overlap the source s is therefore considered for feedback collection (condition: *or* oL in line 12).

The next step is for each source contributing to the solution or for each non-contributing source that has some significant overlap with sources contributing to the solution (line 14), to be added to the set S' which holds the sources on

which feedback needs to be collected (line 15). S' is used in the next cycle to compute a new sample size sS over the remaining unlabelled data items. After a few rounds of feedback collection the scenario can be as in Fig. 2(a), where there is still some overlapping but the sources contributing to the solution are now mostly identified.

The iteration continues while any of the following conditions hold (line 2): (i) There is overlapping between confidence intervals of sources that contribute to the solution and sources that do not contribute. (ii) The number of data items on which to collect additional feedback obtained by using Eq. 9 is greater than zero. In other words, we have still some data items left for feedback collection. (iii) The remaining budget b is greater than zero.

When the loop exits, the solution $\mathbf{X}$ (line 19) is in the form of a collection of counts of the number of data items to be used from each candidate source in S. Figure 2(b) presents a potential image at this stage, where no overlapping exists between confidence intervals of sources contributing and not contributing to the solution. Note that when the loop exists because there is no longer any overlap between the confidence intervals of sources that contribute to the solution and sources that do not contribute, this indicates that the selected sources should not change if additional feedback is collected. This is, thus, a well-founded approach to deciding when additional feedback is unlikely to be fruitful.

6 Evaluation: TFC vs. Random and Uncertainty Sampling

In this section we present the experimental results for evaluating the TFC strategy against two competitors: random and uncertainty sampling. Random acts as a baseline. Uncertainty sampling is a general technique that is applicable to the setting we are exploring. To the best of our knowledge there are no specific contributed solutions to this problem in the research literature.

The *random* sampling does not target specific data items for feedback. This baseline competitor considers, as candidates for feedback, all unlabelled items produced by the sources, providing an even distribution of the feedback collected.

Uncertainty sampling is a technique that follows the active learning paradigm, which is based on the hypothesis that if a learning algorithm is allowed to choose the information from which it is learning then it will perform better and with less training [20]. In the uncertainty sampling technique, an active learner poses questions to an oracle over the instances for which it is less certain of the correct label [11]. Often the uncertainty is represented by a probabilistic model that represents the degree of uncertainty we have in the instances. In this paper, the uncertainty is represented by a heuristic that considers the weights of the criteria and the margins of error for the estimated criterion of a source. Feedback is collected first on those data items whose originating source has the largest margin of weighted error, thus taking into account the importance the user places on the criterion. The uncertainty is computed using this formula

$$u_t = \max(w(\hat{c}_{k_s}) \cdot e(s_t, \hat{c}_{k_s})); k = 1, 2, \ldots, n \tag{17}$$

where t is a data item produced by the source s, u is the uncertainty value on which the items are ranked, w is the data criterion weight, e is the margin of error (Eq. (4)), $\hat{c}_{k_s}$ is the data criterion, and n is the number of criteria. For feedback collection we target first those items with the highest uncertainty, considering all criteria and candidate sources.

6.1 Experimental Setup

The evaluation uses a data set about food (world.openfoodfacts.org/data). This data set contains nutritional information about world food products in an open database format. The information has been gradually collected from vendor's websites and added to the database by unpaid contributors. An additional data set (about UK real estate data) was used to evaluate our approach but since the results were very similar to those presented here, we have not included them as well.

For these experiments, we consider 86,864 different data items produced by 117 virtual sources (where each virtual source represents a contributor to the database). Each data item has the following attributes: code, url, creator, product_name, quantity, origins, countries, serving_size, additives, nutrition_score; all of which were stored as text strings. The targeting approaches were tested with 2, 4 and 6 data criteria, and varied weights among them (user's preferences). The data criteria considered were (in order): correctness, relevance, usefulness, consistency, conciseness and interpretability. The weights corresponding to each tested scenario and for each data criterion are presented in Table 1.

Table 1. Criteria weights for experimental scenarios

	2 criteria ($\mathbf{w}_1$)	4 criteria ($\mathbf{w}_2$)	6 criteria ($\mathbf{w}_3$)
Accuracy (c_1)	0.5	0.4	0.3
Relevance (c_2)	0.0	0.2	0.1
Usefulness (c_3)	0.0	0.1	0.2
Consistency (c_4)	0.5	0.3	0.1
Conciseness (c_5)	0.0	0.0	0.2
Interpretability (c_6)	0.0	0.0	0.1

The experiments were repeated 20 times to reduce the fluctuations due to the random sampling, and the average values were reported. All the statistical computations assume that the data for every source is normally distributed, and are based on a 95% confidence level ($z - score = 1.96$) with an error of 0.05.

For these experiments, the feedback collected is simulated by sampling over the ground truth, in order to evaluate the performance of the approaches without considering additional factors like worker reliability in crowdsourcing [13].

The ground truth was obtained by modelling a typical user's intention over the food data and evaluating this intention across the 6 data criteria in Table 1.

In these experiments, we evaluate the maximum overall weighted utility *oU* by applying the Min-sum model from Eq. 16 to solve the MCSS problem. *oU* is a measure of the utility of a solution considering the user's preferences.

6.2 Results

The plots presented in Fig. 4 show the *oU* for the 3 targeting strategies compared on 3 different scenarios for which the weights are given in Table 1.

In Fig. 4(a) we compare the averaged *oU* for the 3 targeting strategies with incremental levels of feedback for 2 criteria. The dotted line represents a reference solution achieved *without uncertainty* (100% of the data items labelled). In these results, TFC found a solution above 0.8 *oU* with only 2.5% of the data items labelled, in comparison with 0.32 and 0.43 *oU* achieved by the random and uncertainty sampling approaches, respectively, for the percentage of feedback collected. As this scenario considers only 2 criteria ($\mathbf{W}_1$) the solution is hard to find, because the number of potential solutions is larger than when we have more criteria to balance, in other words, if the number of constraints increases (by having more criteria to select the data items), the number of potential solutions decreases which, in turn, reduces the complexity of the optimisation problem.

In Fig. 4(b) TFC still clearly outperforms its competitors, in a scenario with 4 data criteria. The averaged overall weighted utility *oU* for the reference solution is not as high as in the previous scenario due to the reduction in the number of potential solutions, which is caused by imposing more restrictions (more criteria) in the optimisation problem. This reduces the difference between the 3 strategies but, even so, TFC reaches the reference solution with 6% of the data items labelled while the other approaches have reached barely above the half of the reference *oU* at the same point. In terms of the improvement, by using TFC the solution, with 2.5% of labelled data items, has 0.7 *oU*, while random and uncertainty sampling achieve 0.33 and 0.39 respectively.

Figure 4(c) shows the averaged *oU* for the scenario with 6 criteria. In this case, as we increase even more the number of constraints, the optimisation algorithm finds solutions with lower combined *oU* hence the smaller difference between the 3 strategies. The advantage of the TFC approach is smaller but it still reaches the reference solution with less feedback than the competitors. For instance, with 2% of labelled data items, TFC allows a solution with *oU* of 0.6, compared with 0.31 and 0.39 for random and uncertainty sampling respectively.

In the three figures described, the return on investment is clearly favourable for the TFC approach as the overall weighted utility *oU* of the solution achieved by solving the MCSS problem is always larger with TFC, particularly for small amounts of feedback, which is aligned to the objective of reducing the feedback required to obtain effective when following a pay-as-you-go approach.

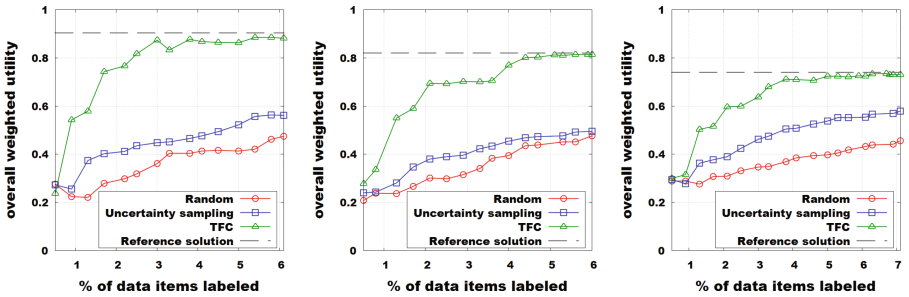

Fig. 4. Results summary for MCSS experiments for (a) 2, (b) 4, and (c) 6 criteria.

7 Conclusions

This paper presented TFC, a strategy for targeting data items for feedback, to enable cost-effective MCSS. TFC was developed to address the problem of incomplete evidence about the criteria that inform source selection. Key features of the approach are that: (i) Feedback is collected in support of multi-criteria optimisation, in a way that takes into account the impact of the uncertainty on the result of the optimisation. (ii) Feedback is collected not for individual sources in isolation, but rather taking into account the fact that the result is a set of sources. (iii) Feedback is collected on diverse types of criteria, of which there may be an arbitrary number, and user preferences in the form of weights are taken into account during the targeting. (iv) Feedback collection stops when the collection of further feedback is expected not to change which sources contribute to a solution (i.e. there is no significant overlap between the criteria estimates for the selected and rejected sources). (v) Experimental results, with real world data, show substantial improvements in the cost-effectiveness of feedback, compared with a baseline (random) solution and an active learning technique. Future work will evaluate the TFC approach using feedback collected using crowdsourcing, and under a complex and well-known problem domain, i.e. supplier selection.

Acknowledgement. Julio César Cortés Ríos is supported by the Mexican National Council for Science and Technology (CONACyT). Data integration research at Manchester is supported by the UK EPSRC, through the VADA Programme Grant.

References

1. Belhajjame, K., Paton, N.W., Embury, S.M., Fernandes, A.A.A., Hedeler, C.: Incrementally improving dataspaces based on user feedback. Inf. Syst. **38**(5), 656–687 (2013)
2. Bozzon, A., Brambilla, M., Ceri, S.: Answering search queries with crowdsearcher. In: WWW 2012, Lyon, France, pp. 1009–1018, 16–20 April 2012
3. Bulmer, M.G.: Principles of Statistics. Dover Publications, New York (1979)

4. Crescenzi, V., Merialdo, P., Qiu, D.: Crowdsourcing large scale wrapper inference. Distrib. Parallel Databases **33**(1), 95–122 (2015)
5. Dong, X.L., Saha, B., Srivastava, D.: Less is more: selecting sources wisely for integration. PVLDB **6**(2), 37–48 (2012)
6. Foley, D.H.: Considerations of sample and feature size. IEEE Trans. Inf. Theor. **18**(5), 618–626 (1972)
7. Franklin, M., Kossmann, D., Kraska, T., Ramesh, S., Xin, R.: Crowddb: answering queries with crowdsourcing. In: ACM SIGMOD, pp. 61–72 (2011)
8. Halevy, A., Korn, F., Noy, N.F., Olston, C., Polyzotis, N., Roy, S., Whang, S.E.: Goods: organizing google's datasets. In: ACM SIGMOD, pp. 795–806 (2016)
9. Hung, N.Q.V., Thang, D.C., Weidlich, M., Aberer, K.: Minimizing efforts in validating crowd answers. In: SIGMOD, Australia, pp. 999–1014 (2015)
10. Knezevic, A.: Overlapping confidence intervals and statistical significance. StatNews, Cornell University, Cornell Statistical Consulting Unit 73 (2008)
11. Lewis, D.D., Gale, W.A.: A sequential algorithm for training text classifiers. In: ACM-SIGIR, pp. 3–12 (1994)
12. Lewis, J.R., Sauro, J.: When 100% really isn't 100%: improving the accuracy of small-sample estimates of completion rates. JUS **3**(1), 136–150 (2006)
13. Liu, X., Lu, M., Ooi, B.C., Shen, Y., Wu, S., Zhang, M.: CDAS: a crowdsourcing data analytics system. PVLDB **5**(10), 1040–1051 (2012)
14. Mozafari, B., Sarkar, P., Franklin, M.J., Jordan, M.I., Madden, S.: Scaling up crowd-sourcing to very large datasets: a case for active learning. PVLDB **8**(2), 125–136 (2014)
15. Pipino, L.L., Lee, Y.W., Wang, R.Y.: Data quality assessment. Commun. ACM **45**(4), 211–218 (2002). Supporting community and building social capital, USA
16. Rekatsinas, T., Deshpande, A., Dong, X.L., Getoor, L., Srivastava, D.: Sourcesight: enabling effective source selection. In: SIGMOD Conference, San Francisco, CA, USA, pp. 2157–2160 (2016). http://doi.acm.org/10.1145/2882903.2899403
17. Rekatsinas, T., Dong, X.L., Getoor, L., Srivastava, D.: Finding quality in quantity: the challenge of discovering valuable sources for integration. In: CIDR (2015)
18. Rekatsinas, T., Dong, X.L., Srivastava, D.: Characterizing and selecting fresh data sources. In: SIGMOD, pp. 919–930 (2014)
19. Ríos, J.C.C., Paton, N.W., Fernandes, A.A.A., Belhajjame, K.: Efficient feedback collection for pay-as-you-go source selection. In: SSDBM, pp. 1:1–1:12 (2016)
20. Settles, B.: Active learning. Synth. Lect. Artif. Intell. Mach. Learn. **6**(1), 1–114 (2012)
21. Ting, S.C., Cho, D.I.: An integrated approach for supplier selection and purchasing decisions. Supply Chain Manag. Int. J. **13**(2), 116–127 (2008)