Design Patterns for Hybrid Algorithmic-Crowdsourcing Workflows

Christoph Lofi
National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-ku
Tokyo 101-8430, Japan
lofi@nii.ac.jp

Kinda El Maarry
Technische Universität Braunschweig
Mühlenpfordtstr. 23
38106 Braunschweig, Germany
elmaarry@ifis.cs.tu-bs.de

***Abstract*— Crowdsourcing has shown to be a powerful technique for overcoming many challenges in data and information processing where current state-of-the-art algorithms are still struggling. This is especially true for workflows that transparently combine algorithmic heuristics and dynamically crowdsourced tasks that are performed by human workers, and which promise to solve even more complex tasks effectively and efficiently. But still, such *hybrid crowdsourcing workflows* can be difficult to approach, and they are often designed in an ad-hoc fashion. Therefore, in this paper, we extensively investigate such crowdsourcing workflows as described in the literature, and abstract *generic design patterns*, which codify commonly recurring challenges and their best-practice solutions. Each design pattern is described and discussed with a special focus on its requirements, constraints, and effects on the overall workflow. We illustrate the practicality of these patterns by providing real-world application examples where such patterns can or have been applied. Furthermore, we showcase how the individual design patterns can be extended and combined to support more complex workflows. Our design patterns provide an extensive overview of the hybrid crowdsourcing workflows' design space, and allow for a more efficient modeling, analysis, and documentation of such workflows.**

Keywords— information processing, crowdsourcing, workflows, design patterns

I. INTRODUCTION

The ability to intelligently process and analyze large data sets is one of the central challenges of current information systems research. While there have been significant advances in almost all areas of algorithmic data processing, state-of-the-art techniques can still frequently fall short. This is especially true when actual cognitive abilities are required, as is the case in many areas like textual sentiment analysis, information extraction, resource classification, or many knowledge engineering tasks. Also, with the increasing complexity of those algorithms, the susceptibility for errors or the danger of over-specialization increases, and many failings can be traced back to limited cognitive abilities, missing contextual knowledge, or simply failing heuristics.

Therefore, crowdsourcing has become a popular approach for many problems that cannot be easily addressed by automated methods and algorithms, or explicitly require significant amounts of human intelligence or human feedback. Instead of using error-prone and still imperfect algorithmic approaches, such tasks are simply outsourced to real humans. Currently, a variety of platforms like Amazon Mechanical Turk, CrowdFlower,com, or ClickWorker.com are offering services with differing degrees of sophistication, where any kind of (usually relatively simple) cognitive tasks can be dynamically posted and processed by a readily available workforce. Those workers are recruited and retained through payments. Of course, crowdsourcing brings its own challenges: with increasing workloads and tasks' size, crowdsourcing can quickly become expensive in terms of money and the required processing time. Furthermore, the quality of the work provided raises many concerns as platforms are continuously challenged by workers with insufficient skillsets to solve a given task, or even workers who are malicious and aim to only exploit the system. Therefore, quality control and efficiency attainment are central issues for tasks relying on crowdsourcing.

While many current state-of-the art systems either use algorithmic-heuristic workflows, or heavily rely on crowdsourcing, we focus in this paper on the currently rising challenge of *hybrid crowd-sourcing systems with algorithms and human workers dynamically cooperating in a combined workflow*. Such hybrid architectures transparently combine the efficiency of current algorithms with the cognitive power and flexibility of human beings. Hence, they bridge the semantic gap in today's information processing for increased *efficiency* and *effectiveness*.

Basically, two major approaches to such hybrid system designs can be identified:

- Using human input to improve information processing algorithms' performance by providing training samples, answering questions about ambiguous results, or by providing relevance feedback
- Involving humans directly into the information processing process by explicitly outsourcing some of the required tasks or operators within the process

Designing workflows that use such hybrid information processing is challenging and quite often a complex task. Moreover, it's often tailored for a specific use case and in an ad-hoc fashion. Also, as such approaches are still very new, little structured research on this topic has been established yet. Therefore, in this paper we examine frequently reoccurring

design decisions in such hybrid crowd-sourcing workflows, and abstract them into easy to understand and to combine *design patterns*, which allow for a structured approach towards designing such processes. Our contributions are as follows:

- We motivate common scenario for hybrid crowdsourcing workflows, highlighting and detailing their challenges and their importance for information systems research.
- We provide an overview over existing classification schemes and patterns for general crowd-sourcing processes.
- As our main contribution, we introduce a set of design patterns for hybrid crowd-sourcing workflows, and discuss their applicability, effects on result quality and costs, and outline their use-cases.

II. Hybrid Crowd-sourcing

In the first incarnations of the term crowdsourcing by Jeff Howe in 2006 [1], it was understood as "the act of a company or institution taking a function once performed by employees and outsourcing it to an undefined (and generally large) network of people in the form of an open call." Boosted by the rise of the Social Web and the general spread of internet usage around the world, crowdsourcing gained significant momentum and this label has been applied to a wide variety of different platforms. While also collaboratively created artifacts like Wikipedia or open-source software can be considered as a result of crowdsourcing, in this paper, we restrict our view to crowdsourcing as outsourcing information processing operators of an information system to human workers, who are recruited and retained via the Web. This includes platforms like Amazon.com or TripAdvisor.com, who motivate their users to provide detailed reviews and ratings for the services and products offered by the platform. These reviews and ratings are then in turn used for better query processing and personalization, leading to increased revenue. But more specifically, this includes generic crowdsourcing platforms that are open to anyone like Amazon Mechanical Turk or CrowdFlower.com. Such platforms allow any large task to be executed by dividing it into many small and simple tasks (called HITs, Human Intelligence Tasks; the smallest unit of crowdsourceable work). These HITs are then distributed to the available human worker pool. Workers are recruited and retained through payments, and hence, these platforms could theoretically be used to perform any given dividable task that requires human intelligence. A central challenge when using such services is how to design information processing workflows that incorporates crowdsourcing, especially with the monetary costs and quality of results in mind. Recently, the Amazon Mechanical Turk team posited that extensive research for establishing a library of standard design patterns for HIT design, workflow design, and reviewing methodologies would benefit the crowdsourcing field tremendously and would increase the work quality, leading to better and more consistent results [2]. But still, there is only little research discussing these issues.

Despite the expected diversity of crowdsourcing tasks, the requirements of many businesses are quite repetitive. For instance, CrowdFlower.com discovered that many of their customers performed only one of 20 different reoccurring crowdsourcing tasks, for which it consequently introduced respective user interface templates. These cover mostly very basic data completion or data analysis tasks as for example opinion mining of Social media posts for market analysis, labeling and filtering of images or text, discovery and completion of (business) data sets, forum and community moderation, or user surveys. Here, it is especially interesting that for many of these problems, purely algorithmic approaches, which could process even large tasks for significantly lower costs than human workers are readily available. However, their accuracy and reliability is often not yet suitable for many quality-conscious productive environments (as for example for sentiment analysis [3] or emotion analysis [4] in text, for image moderation [5], or for text summarization and keyword extraction [6]). This shows that there are great opportunities in these areas for hybrid algorithms, which combine the efficiency of these algorithmic approaches with a carefully targeted utilization of the cognitive capabilities of human workers, who can be recruited via crowdsourcing.

To illustrate the potential of these hybrid approaches, consider the experiment presented in [7]. Here, a large collection of movies is to be annotated with their respective genre labels. Assuming that this metadata is not readily available in other data sources, providing those labels becomes a surprisingly expensive and laborious task, as it requires annotators to know the movies, or to be reliably able to infer the genre from other meta-data. Obvious approaches to this challenge would be to either hire in-house "movie experts" to provide those labels manually, or to crowdsource the task to platforms like Amazon Mechanical Turk (which is often cheaper and shows lower overhead). On the other hand, crowdsourcing has to also deal with low quality or malicious results. This is often controlled through applying majority voting between a large numbers of worker judgments. Workers who repeatedly fail to answer "Gold" test questions correctly are then excluded. In any case, monetary costs and time required linearly increase with the number of movies, and continuously increase with higher quality requirements (as more judgments per movie are required for majority voting). But still, for some tasks not even relying on a large number human judgments can provide satisfying quality. In an experiment with 1000 movies, the crowd was asked to judge whether each movie was a comedy or not. Even with 10 judgments per movie, only 64% of the movies were correctly classified, incurring a cost of $20 USD. However, when a hybrid process is employed, higher quality can be achieved at significantly lower costs: in another experiment, a workflow employed crowdsourcing to train and boost a machine learning classifier. The learning classifier operated on a high-dimensional so-called perceptual space, which was created by aggregating simple crowd-judgments like star ratings. Interestingly, the same quality as reported in the first approach was reached by just spending $2.80. While spending the same amount of $20, boosted the quality up to 80%. Furthermore, the costs for this technique are nearly independent from the size of the input data, and even significantly larger movie collections can be labeled by only slightly increasing costs. This example clearly demonstrates the potential of hybrid crowdsourcing workflows.

III. TOWARDS DEFINING CROWDSOURING DESIGN PATTENRS

In this section we overview different classification schemes that were proposed for crowdsourcing. Many of these focus on crowdsourcing-only workflows, and rarely concentrate on the more powerful hybrid fusion of both algorithmic and crowd-based approaches. Still, this overview helps to illustrate the broad range of applications where crowdsourcing can be applied, and serves as a basis for our hybrid crowdsourcing design patterns' abstraction.

A. Crowdsourcing Classification schemes and Taxonomies

In academic research, different classification schemes and taxonomies for crowdsourcing tasks have been investigated from the perspective of different fields like Information Systems [8], business management [9], and HCI (Human Computer interface) [10]. In contrast to what we aim for in this work, many of these classification schemes are based on identifying dimensions or characteristics of general crowdsourcing processes. These studies were mostly motivated by the desire to understand the fine details of the crowdsourcing phenomenon, and accordingly support the management of the crowdsourcing processes. Such classification schemes support organizations, which are interested in crowdsourcing, to better understand the nature of their tasks. This eventually leads to better decision-making and better-performing crowdsourcing processes. For instance, in [11], a classification scheme is proposed after analyzing 46 real-life examples described in the literature (e.g. in-house crowdsourcing as used by TripAdvisor, iStockphoto, etc.). The following four dimensions for differentiating crowdsourcing processes were identified: 1) potential for pre-selecting most suitable contributors, 2) degree of peer contributions' disclosure i.e. the extent to which contributors can access each other's contributions, 3) type of result aggregation, and 4) type of contributors' compensation. Similarly, [12] classifies the crowdsourcing processes based on three dimensions 1) nature of the task, 2) nature of the crowd, and 3) nature of the payment.

Another proposed classification scheme [9] distinguishes the broad classes of crowdsourcing practices, by differentiating between the type of tasks that are crowdsourced (simple, complex and creative tasks), and the nature of the crowdsourcing process: 1) selective process that's used to find a single candidate solution for a specific need, or 2) integrative process that's used to build information bases. In such a process individual contributions are aggregated and have little individual impact.

A different take on classifying crowdsourcing [13] considers its varying applications: product development and configuration, products designs, product rating, etc.

Though these classification schemes may give some insights to organizations wishing to venture into crowdsourcing and to invest in it as an innovative business solution, none of them cover how to effectively design functional crowdsourcing workflows from a technical perspective.

B. Design Patterns

In software engineering, a design pattern is defined as a general reusable solution to a commonly occurring problem [14]. It serves as a template or a description of how to solve a problem. A Pattern is defined by: 1) its name, 2) the set of problems it is applicable to 3) a description of the solution, which is often illustrated by an abstract architecture, model, or generic code, and 4) its entailing consequences, which encompasses both results and tradeoffs.

Developing design patterns that capture past experiences and best practices would support organizations to design more stable and effective crowdsourcing workflows. Moreover, such patterns could quickly be applied without the need to reinvent the wheel. With such flexible and reusable patterns, minimum redesign is required if not completely avoided.

C. Workflow Patterns in Crowdsourcing

As already used in software engineering, workflow patterns are a specialized form of design patterns [15]. In particular, workflow patterns are related to the development of workflow application and in a broader sense to the development of process-oriented applications [16]. Some limited work on workflow patterns for crowdsourcing has already been performed. One study that focused on collaborative writing, brainstorming, and transcription problems [17] identified two approaches based on how workers contribute and collaborate. Here, workers can either work in parallel or in an iterative fashion, where their contribution builds on top of each other's. Experimental results supported an iterative workflow pattern for refining tasks (e.g. writing, collective brainstorming), which increases the average quality. On the other hand, parallel workflow patterns were recommended for creative tasks (e.g. transcribing blurry texts, individual brainstorming), where a higher variability of responses is more beneficial. In [18], a workflow pattern for improving results was introduced with a focus on proofreading and editing text. The pattern splits the proofreading task into a series of generation and review stages, which recruit workers to find candidate text areas that can be improved, then collect a set of candidate modifications, and finally filter out bad ones.

IV. HYBRID CROWDSOURCING DESIGN PATTERNS

Identifying commonly reoccurring challenges and solutions, we propose in this section a set of workflow design patterns for crowdsourcing. We focus on hybrid crowdsourcing, i.e. those approaches where the advantages of both algorithmic and crowd-based systems are to be fused. Keeping in-style with design patterns as used in software engineering, we document each pattern by its motivation, the requirements for its applicability, a description of its structural design, the consequences of applying it, and a brief survey of its known uses as found in literature or in real-life. Furthermore, the presented patterns can also be combined into more elaborative workflows, as described in section V.

These patterns are intended to support the design and especially the documentation of hybrid crowdsourcing workflows, and to serve as a communication tool for clarifying workflows. Therefore, they are not necessarily disjoint or minimal, but are chosen to highlight certain motivations or design decisions. Furthermore, due to space limitations, the following list of patterns are not exhaustive as we limited ourselves to the more interesting patterns, leaving out well-known or overly trivial ones (like performing majority votes to increase result quality of a purely crowd-based workflow).

A. Pattern: Magic Filter

1) Motivation

Though crowdsourcing can be an attractive solution, employing it in an unrestrictive manner incurs unnecessary monetary and time costs. Therefore, it can be worthwhile to limit the number of HITs to the most relevant ones. An algorithmic filter is employed for this decision, which classifies the data points to be either crowdsourced or ignored.

2) Applicability Requirements

For this pattern to be applicable, it is important that the original crowdsourcing tasks are overwhelmingly large and contain some data points that are less important or even irrelevant to the overall result, and some that are more relevant. Often, algorithmic filters for identifying irrelevant data are based on some features exhibited by the data, which can be used to decide whether crowdsourcing is needed or not. The feature upon which the filtering mechanism is based on should be highly discriminative i.e. only and reliably filter out the data not requiring any crowdsourcing efforts.

3) Structural design description

Fig. 1. Magic Filter Design Pattern

Magic Filter is one of the simplest hybrid crowdsourcing patterns. Given a dataset from which the HITs will be formulated and issued to the crowd, a preprocessing step is carried out. By carefully analysing the data, an algorithm that acts as a filter can be deployed as a feasible solution to narrow down the number of HITs. As illustrated in figure 1, only this reduced data is then used to formulate the HITs to be issued to the crowd.

4) Consequences

By inserting a filter in the workflow as a preprocessing step, the number of HITs are reduced. Consequently, this decreases both time and monetary costs that the crowdsourcing process incurs by not processing all of the data points (which needs to be acceptable). As long as the filter is working as intended and filters out only those data points where human feedback is unnecessary, the overall result quality is not reduced by this process. On the other hand, if the feature(s) upon which the filter bases its decision isn't discriminative enough, then the overall quality can degrade, especially with the introduction of false negatives (i.e. data points that were filtered out by mistake). Depending on the use case, false negatives could then mean data entries with still incomplete data, images that aren't tagged, mistranslated sentences that aren't fixed, etc.

5) Example Applications

In [19], this technique is applied to a continuous video stream of underwater footage, where fish species are to be tagged. Given the extremely large number of video frames, not every frame needs to be tagged. Therefore, frames are first clustered, and only some representatives of each cluster are crowdsourced for manual classification. Similarly, for a use case that uses crowdsourcing to detect crimes in security footage, an algorithmic filter could exclude all video sequences where there are no people in the scene, or all people behaving "normal", or are not close to security-relevant objects.

B. Pattern: Crowd Trainer

1) Motivation

Algorithms that are based on machine learning, like for instance many data classification algorithms, can often perform their tasks rather reliably. However, they require extensive training, and training any learning algorithm requires an annotated training dataset. Unfortunately, the acquisition of such datasets is expensive and time-consuming, and therefore often only few labels per data point are used which limit the full potential of many algorithms. Also, in many companies, it is still very common to use in-house employees or specifically hired annotators for the task of creating training sets. In the light of these limitations, dynamically recruiting human workers via crowdsourcing to label these datasets in bulk, at cheap costs, and with fast completion rates becomes rather favourable [20]. This also allows for better training sets as more judgments/labels per data point can be incorporated for the same price.

2) Applicability Requirements

For the task to be solved, there needs to be an available machine learning based algorithm, which if trained correctly, provides indeed the required result quality. Furthermore, crowd workers must be able to provide training data with satisfying quality.

3) Structural design description

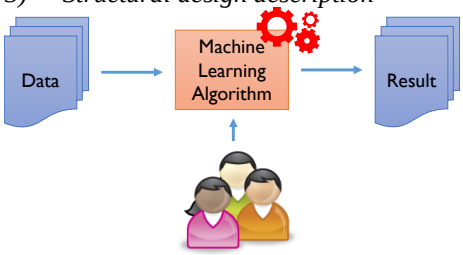

Fig. 2. Crowd Trainer Design Pattern

As depicted in figure 2, given an untrained supervised machine learning algorithm, the unannotated training dataset is crowdsourced to acquire the correct labels. The crowds' judgments can be then used to train the learning algorithm, which enables it in the future to process or classify all data at no monetary costs.

4) Consequences

By turning to the crowd to annotate datasets for training learning algorithms, this overcomes the limited number of experts who traditionally provided these annotations. Experimental findings even indicate that crowd workers are as good as experts for example in the area of information relevance assessment (as shown by [21] for the TREC[1] relevance challenges). Furthermore, given the relatively cheap costs of crowdsourcing solutions, the size of the training dataset can be

[1] http://trec.nist.gov/ (Text Retrieval Conference)

rather big, which offers more extensive training for learning algorithms.

But still, making the best use of the crowd poses a challenge. With a fixed budget, there is always the trade-off between the number of annotations collected per label and the number of labels to be annotated. Experimental results show that the level of agreement between annotators is central to this trade-off. When there's a high level of agreement between annotators, acquiring multiple annotations for each label ceases to be useful and it is better to direct the crowd's annotation efforts to maximize the size of the training data [22]. Furthermore, the quality of the annotations provided by the crowd also can raise many concerns depending on the scenario. Data quality can be improved through applying different repeated-labeling strategies of increasing complexities (for more details see [23]).

5) Example Applications

Recently, it has become a common practice for training classifier algorithms to turn to the non-expert workforce made available through the various crowdsourcing platforms. For example, the crowd have been providing sentiment annotations [24], activity recognition annotations [25], spam detection, etc. But under certain conditions this pattern can also be applied to non-obvious applications as for example for completing incomplete database entries, as in [7].

Other applications where this pattern is well suited is machine translation [26]. Automatic translation techniques as e.g. example-based translation, SMT (Statistical Machine Translation), or corpus based approaches require large volumes of training data, so called parallel corpora. In [27], active learning approaches is employed to identify those sentences, which would be the most informative training examples for the translation systems to learn from. Furthermore, quality control can be exercised to assure near professional level translation acquisition from the crowd. A set of features were for instance proposed in [28], which model both the translations and translators. Based on these features, submitted translations can be scored to reliably distinguish the good from the bad acquired translations.

Relevance feedback represents yet another viable application for the *Crowd Trainer* pattern. Relevance feedback is typically used for enhancing information retrieval algorithms, especially for learning the best result rankings. Other relevance evaluation approaches include click-through data analysis and query logs. Recent studies have already started exploiting crowdsourcing for relevance feedback. For instance, [29] introduces an evaluation approach that crowdsources small evaluation tasks as a complimentary alternative to traditional relevance evaluation approaches.

C. Pattern: Machine Improvement

1) Motivation

Though many state-of the-art algorithms are very efficient at solving problems at exceedingly high speeds, they don't often lead to high quality results, especially in cases where some human intelligence or creativity is required. When the result's quality is good enough, improving the imperfect algorithmic results using crowdsourcing is significantly easier and cheaper than asking the crowd to reach the result directly from scratch. This situation is exploited in this pattern.

2) Applicability Requirements

An algorithmic approach that solves the problem is required, even if its produces insufficient quality-wise results. However, it must be significantly easier for human workers to improve this imperfect result compared to solving the task completely on their own.

3) Structural Design Description

Fig. 3. Machine Improvement Design Pattern

As depicted in Figure 3, the algorithmic heuristic in place initially processes the data and computes the results. The outputted low quality results are then formulated to form HITs, which are then sent to the crowd.

4) Consequences

High quality results can be achieved rather cheaply by only adding little human intelligence to the algorithmic results. Often, quality can also be significantly higher compared to a crowdsourcing-only workflow, at additionally lower costs [30].

5) Example Applications

A prevalent application for the *Machine Improvement* design pattern is again machine translation. Even a well-trained machine translation engine (trained using e.g. the *Crowd Trainer* pattern) do often not provide perfect results yet. On the other hand, for a skilled human it is comparably easy to refine the suggested translations very quickly. Therefore, the automatically translated text is directly crowdsourced for improvements. Similar to the work flow introduced for proof-editing in [18], breaking the final crowdsourcing workflow into several cascading workflows with increasing granularity often yields better results. For example, starting off with a task where the workers are to highlight parts of the text requiring improvements, this can be followed by another task asking for the corresponding modifications for the previously flagged text parts.

The *Machine Improvement* pattern can also be applied to Optical Character Recognition (OCR) tasks. While deciphering a low quality text image is a trivial but expensive task for humans, it's much more complicated but significantly faster and cheaper for machines. Involving humans in the process presents itself as an intuitive solution for improving the OCR algorithms' output. A prime example for this approach's success is the National Library of Australia, which employs a crowdsourcing workforce to improve the electronically translated OCR text of old newspapers. Nearly 2 years after the launch of the project, 12 million lines of text were improved for surprisingly low costs [31], and many other libraries followed in their footsteps.

D. Pattern: Virtual Worker

1) Motivation

Aggregating judgments of different workers for the same HIT is an integral part of crowdsourcing quality management. By combining the results of multiple users, who are influenced

by their own unique experiences, points of view, and skillsets, even simple aggregation techniques like majority votes lead to significantly improved overall quality and can even cancel out malicious users. A similar reasoning is also applied in purely algorithmic environments, where different heuristics, which are based on different assumptions and features, are executed in parallel and their individual results are combined in a final step. Again, the assumption is that the different approaches will cancel out the individual weaknesses upon final output aggregation. In the *Virtual Worker* pattern, the judgments of human workers and those of heuristic approaches are aggregated in order to decrease the costs as opposed to to purely crowdsourcing based approaches, and to increase the variety and richness of judgments as opposed to machine-only based approaches. At its core, the *Virtual Worker* pattern aims at combining multiple weak responses to eventually achieve a higher quality-wise result.

2) Applicability Requirements

For this pattern to be applicable, aggregating multiple judgments should be possible. Also, this pattern is particularly effective for tasks where even human workers have problems reaching a consensus. By aggregating the individual judgments, the quality of the results increases (e.g. as often is the case with classification or labeling). Furthermore, at least one suitable heuristic for providing heuristic judgments is required, and its quality should not be much worse than that of average humans.

3) Structural design description

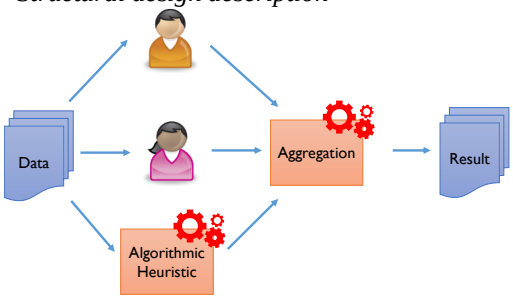

Fig. 4. Virtual Worker Design Pattern

In the *Virtual Worker* pattern, the judgments of both humans and heuristics are aggregated into a final judgment (i.e. heuristics can transparently replace workers in a crowdsourcing aggregation process). For aggregating judgments, majority votes, weighted votes, or different averaging operators can be commonly used.

4) Consequences

Due to the higher variety and number of judgments, usually aggregated results should have higher quality. Furthermore, judgments attained through heuristics do not incur additional crowdsourcing costs.

5) Example Applications

Usually, aggregating multiple judgments is only performed for aggregating different heuristic approaches (e.g., boosted learning [32], which combines multiple weak learning algorithms, multi-classifier classification [33], or multi-heuristic question answering [34]), or for aggregating several crowd judgments as a standard tool for quality management e.g., majority votes – as of yet hybrid combinations are rarely to be found despite their potential power. In [35], the challenging problem of solving SAT analogy challenges for college admission using crowdsourcing was examined. Even though individual workers showed low average quality (only 55% of all challenges answered correctly), weighted aggregating increased to over 80%. Heuristics for the same problem show similar quality as human workers. Furthermore, a combined hybrid approach was suggested as being very promising.

E. Pattern: High Confidence Switching

1) Motivation

Not only can some machine learning or heuristic applications solve a problem, but they can also provide a confidence estimate on how accurate and reliable the solution is likely to be e.g. very common for medical diagnosis, which often comes with a risk error assessment. For these algorithms, also known as confidence machines [33] [35] [36], predictions with high confidence are naturally more preferable to those with low confidence values. With this pattern, there is no need to accept solutions with low confidence levels. Instead, based on the confidence value of the algorithmic solution, the workflow can simply retain high confidence results, and switch for low confidence results to different post-processing steps like replacing or improving low confidence values with more reliable human judgments.

2) Applicability Requirements

Similar to the *Magic Filter* pattern, the algorithm should be based on a feature from which a reliable confidence value for the corresponding prediction can be computed. These confidence levels should correctly reflect the actual risk associated with their predicted values. While this is easy to achieve in some domains, it might be hard in others.

3) Structural design description

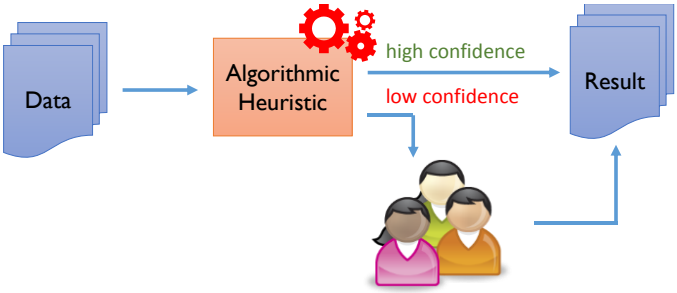

Fig. 5. High Confidence Switching Design Pattern

Figure 5 illustrates the typical flow of *High Confidence Switching* pattern. Initially, an algorithm processes the data and computes both the output along with its corresponding confidence level. Results with high confidence instantly constitute part of the final results, while those with low confidence are sent to the crowd to complement the low confidence with the crowds' insight.

4) Consequences

The hybrid solution of incorporating humans with confidence machines, improves the efficiency and reliability of these algorithms for cases where the algorithm fails to make

a prediction of high confidence. By exploiting the confidence values that are computed, only part of the predictions can be crowdsourced.

5) Example Applications

A good example of the *High Confidence Switching* pattern in practice can be illustrated with entity resolution algorithms, which are used to match pairs of records that might refer to the same entity in a database system. In [37], a two-tiered heuristic approach is proposed, where an algorithm passes over the data, computing for each pair a matching likelihood. Based on this likelihood, pairs with low matching confidence are reviewed by the crowd for verification, while those matching with high confidence become conclusive. Experimental results backs up the attained efficiency and high accuracy.

Another successful application showcasing the potentials of the *High Confidence Switching* pattern can be found in crowd-enabled databases. Crowd-enabled databases effectively deals with the widespread problem of incomplete data during runtime by dynamically completing these values through crowdsourcing tasks. Similar to confidence machines, a heuristic is proposed in both [38] and [39], which optimizes the performance of skyline queries when hampered by the presence of incomplete tuples. This is attained through predicting the incomplete values and assessing the individual risk an incomplete tuple poses to the overall resulting quality, rather than the confidence of the predicted value. Only those incomplete tuples with a highly computed risk of degenerating the expected quality are crowdsourced, while the others are completed with the predicted values.

V. Crowdsourcing Design Patterns in Combination

In the previous section, we introduced five design patterns for commonly re-occurring hybrid crowdsourcing solutions. Even though each of the design patterns was presented as a standalone solution, they're not only limited to that. For more complicated workflows, the different design patterns are extensible and can be flexibly combined together to form more elaborate patterns. For illustration purposes and without loss of generality, consider the combination in figure 6. In this workflow, three different design patterns were deployed: *Crowd Trainer*, *Machine Improvement* and *High Confidence Switching*. Initially, following the *Crowd Trainer* pattern, a machine learning based algorithm is trained by annotated datasets provided by the crowd. As soon as the training phase ends and the algorithm starts processing real data points, new design patterns come in to play as the algorithm starts producing results of varying quality. As explained in the *High Confidence Switching* pattern, based on the confidence value the algorithm assigns to the result, the result can be either considered safe and directly used or can be associated with a high risk factor.

The workflow below is implemented by the following example task: consider a workflow for recognizing text in scanned documents. Assuming that the documents use a rare font face that is not supported by off-the shelf software (for example as found in historic printed documents), a machine learning-based OCR algorithm can be trained using the *Crowd Trainer* pattern. Next, the algorithm can be used to transcribe all texts. Often, this kind of algorithm can also provide a confidence value for every recognized word or sentence. These

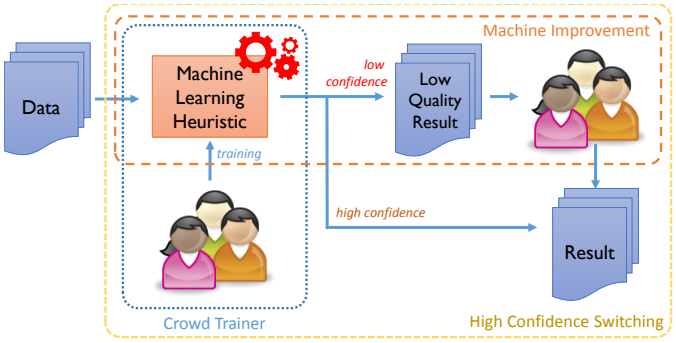

Fig 6. Combination of three Patterns

values can then be used to filter out all text fragments that are likely to be erroneous using the *High Confidence Switching* pattern, while directly retain those fragments showing sufficiently high quality. As per the *Machine Improvement* pattern, the erroneous fragments are then forwarded to the crowd for further improvement. As a final result, this workflow can approach the task of recognizing text over even large document collections rather efficiently and at low monetary costs.

VI. Conclusion

In this paper we focused on hybrid crowd-sourcing systems that combine the dynamic cooperation of both algorithms and human workers. Such systems get the best of both worlds: the efficiency of the algorithms and the cognitive power and insight of humans. Despite the expected diversity of crowdsourcing tasks, many of the requirements are quite repetitive and similar. Examining these frequently reoccurring processes, we abstracted five design patterns to promote a more structured approach towards designing such processes. We showcased the applicability of the design patterns in practice as well as their rewarding effects on the result's quality and cost, underlining this with numerous examples as found in the literature. Furthermore, for more complicated workflows, these design patterns can be treated as building blocks that can be combined together for creating more elaborate patterns.

We don't consider these design patterns to be complete, and surely as crowdsourcing continues to thrive, more applications are bound to appear and exploit the power provided by the crowd in more innovative ways. Continuous observation of how the crowdsourcing applications are evolving will shape up more news patterns.

Acknowledgment

This research has been funded by the FIT-Worldwide program of the German Academic Exchange Service (DAAD).

REFERENCES

[1] J. Howard, "The Rise of Crowdsourcing," *Wired*, vol. 14, 2006.

[2] J. J. Chen, N. J. Menezes, A. D. Bradley, and T. A. North, "Opportunities for Crowdsourcing Research on Amazon Mechanical Turk," *Hum. Factors*, vol. 5, p. 3, 2011.

[3] B. Liu, "Sentiment Analysis and Opinion Mining," *Synth. Lect. Hum. Lang. Technol.*, 2012.

[4] A. Agrawal and A. An, "Unsupervised Emotion Detection from Text using Semantic and Syntactic Relations," in *Int. Conf. on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, 2012.

[5] J. A. Marcial-Basilio, G. Aguilar-Torres, G. Sánchez-Pérez, L. K. Toscano-Medina, and H. M. Pérez-Meana, "Detection of pornographic digital images," *Int. J. Comput.*, vol. 2, pp. 298–305, 2010.

[6] G. Ercan and I. Cicekli, "Using lexical chains for keyword extraction," *Inf. Process. Manag.*, vol. 43, no. 6, pp. 1705–1714, 2007.

[7] J. Selke, C. Lofi, and W.-T. Balke, "Pushing the Boundaries of Crowd-Enabled Databases with Query-Driven Schema Expansion," in *38th Int. Conf. on Very Large Data Bases (VLDB)*, 2012, pp. 538–549.

[8] V. Zwass, "Co-Creation: Toward a Taxonomy and an Integrated Research Perspective.pdf," 2010. .

[9] E. Schenk and C. Guittard, "Towards a characterization of crowdsourcing practices," *Journal of Innovation Economics*, vol. 7, no. 1. p. 93, 2011.

[10] A. J. Quinn and B. B. Bederson, "Human computation: a survey and taxonomy of a growing field," in *Proceedings of the 2011 annual conference on Human factors in computing systems - CHI '11*, 2011, pp. 1403–1412.

[11] D. Geiger and S. Seedorf, "Managing the Crowd: Towards a Taxonomy of Crowdsourcing Processes.," …, pp. 1–15, 2011.

[12] A. Rouse, "A Preliminary Taxonomy of Crowdsourcing," *ACIS 2010 Proc.*, p. 76, 2010.

[13] F. Kleemann, G. Voß, and K. Rieder, "Un (der) paid Innovators: The Commercial Utilization of Consumer Work through Crowdsourcing," *Sci. Technol. Innov. Stud.*, vol. 4, no. 1, pp. 1–22, 2008.

[14] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design patterns: elements of reusable object-oriented software*. Addison-Wesley, 1995, p. 395.

[15] M. S. Saravanan and R. J. Rama Sree, "Workflow driven Process controlling for unstructured activities," *European Journal of Business and Management*, 2011. .

[16] W. van der Aalst and A. H. M. ter Hofstede, "Workflow Patterns," *Distrib. Parallel Databases*, vol. 14, no. 1, pp. 5–51, 2003.

[17] G. Little, "Exploring iterative and parallel human computation processes," *Proc. 28th Int. Conf. Ext. Abstr. Hum. factors Comput. Syst. CHI EA 10*, p. 4309, 2010.

[18] M. S. Bernstein, G. Little, R. C. Miller, B. Hartmann, M. S. Ackerman, D. R. Karger, D. Crowell, and K. Panovich, "Soylent: a word processor with a crowd inside," in *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, 2010, pp. 313–322.

[19] J. He, J. van Ossenbruggen, and A. P. de Vries, "Do you need experts in the crowd?: a case study in image annotation for marine biology," in *Conference on Open Research Areas in Information Retrieval*, 2013, pp. 57–60.

[20] P. Hsueh, P. Melville, and V. Sindhwani, "Data quality from crowdsourcing: a study of annotation selection criteria," *Proc. NAACL HLT 2009 Work. Act. Learn. Nat. Lang. Process.*, no. June, pp. 27–35, 2009.

[21] O. Alonso and R. Baeza-Yates, "Design and implementation of relevance assessments using crowdsourcing," in *Proceedings of the 33rd European conference on Advances in information retrieval*, 2011, pp. 153–164.

[22] A. Brew, D. Greene, and P. Cunningham, "The Interaction Between Supervised Learning and Crowdsourcing.pdf," *NIPS workshop on computational social science and the wisdom of crowds*, 2010. .

[23] V. S. Sheng, F. Provost, and P. G. Ipeirotis, "Get another label? improving data quality and data mining using multiple, noisy labelers," *New York*, pp. 614–622, 2008.

[24] A. Brew and D. Greene, "Using Crowdsourcing and Active Learning to Track Sentiment in Online Media," in *Proceedings of the 2010 conference on ECAI 2010: 19th European Conference on Artificial Intelligence*, 2010, pp. 145–150.

[25] L. Zhao, G. Sukthankar, and R. Sukthankar, "Incremental Relabeling for Active Learning with Noisy Crowdsourced Annotations," *2011 IEEE Third Int'l Conf. Privacy, Secur. Risk Trust 2011 IEEE Third Int'l Conf. Soc. Comput.*, pp. 728–733, 2011.

[26] A. Kunchukuttan, S. Roy, P. Patel, K. Ladha, S. Gupta, M. Khapra, and P. Bhattacharyya, "Experiences in Resource Generation for Machine Translation through Crowdsourcing," *Lr. 2012 Eighth Int. Conf. Lang. Resour. Eval. 21-27 May 2012, Istanbul, Turkey*, pp. 384–391, 2012.

[27] V. Ambati, S. Vogel, and J. Carbonell, "Active Learning and Crowd-Sourcing for Machine Translation," in *Proceedings of the LREC*, 2010, pp. 2169–2174.

[28] O. F. Zaidan and C. Callison-Burch, "Crowdsourcing Translation: Professional Quality from Non-Professionals.," *ACL*, pp. 1220–1229, 2011.

[29] O. Alonso, D. E. Rose, and B. Stewart, "Crowdsourcing for relevance evaluation," *ACM SIGIR Forum*, vol. 42, no. 2. p. 9, 2008.

[30] T. Aikawa, K. Yamamoto, and H. Isahara, "The Impact of Crowdsourcing Post-editing with the Collaborative Translation Framework," *Adv. Nat. Lang. Process.*, vol. 7614, no. 1–10, 2012.

[31] R. Holley, "Many Hands Make Light Work : Public Collaborative OCR Text Correction in Australian Historic Newspapers - Version details - Trove," *Book*, 2009. .

[32] Y. Freund, R. E. Schapire, and P. Avenue, "A Short Introduction to Boosting," *Society*, vol. 14, no. 5, pp. 771–780, 1999.

[33] R. Ranawana and V. Palade, "Multi-Classifier Systems: Review and a roadmap for developers," *Int. J. Hybrid Intell. Syst.*, vol. 3, no. 1, pp. 35–61, 2006.

[34] D. Ferrucci, E. Brown, J. Chu-Carroll, J. Fan, D. Gondek, A. A. Kalyanpur, A. Lally, J. W. Murdock, E. Nyberg, J. Prager, N. Schlaefer, and C. Welty, "Building Watson: An Overview of the DeepQA Project," *AI Mag.*, vol. 31, no. 3, pp. 59–79, 2010.

[35] C. Lofi, "Just ask a human? – Controlling Quality in Relational Similarity and Analogy Processing using the Crowd," in *CDIM Workshop at Database Systems for Business Technology and Web (BTW)*, 2013.

[36] V. Vovk, "On-line confidence machines are well-calibrated," *43rd Annu. IEEE Symp. Found. Comput. Sci. 2002. Proceedings.*, 2002.

[37] J. Wang, T. Kraska, M. J. Franklin, and J. Feng, "CrowdER: crowdsourcing entity resolution," *Proc. VLDB Endow.*, vol. 5, no. 11, pp. 1483–1494, 2012.

[38] C. Lofi, K. El Maarry, and W.-T. Balke, "Skyline Queries over Incomplete Data - Error Models for Focused Crowd-Sourcing," *ER*, 2013. .

[39] C. Lofi, K. El Maarry, and W.-T. Balke, "Skyline Queries in Crowd-Enabled Databases," *EDBT/ICDT Joint Conference*, 2013. .