Recommender system for crowdsourcing platform Witology

Dmitry I. Ignatov
School of Applied Mathematics and
Information Science
National Research University Higher School of Economics
Moscow, Russia
dignatov@hse.ru

Alexandra Kaminskaya
School of Applied Mathematics and
Information Science
National Research University Higher School of Economics
Moscow, Russia
skam90@gmail.com

Natalia Konstantinova
University of Wolverhampton, UK

Andrey Konstantinov
School of Applied Mathematics and
Information Science
National Research University Higher School of Economics
Moscow, Russia

***Abstract*—This paper discusses the recommender models and methods for crowdsourcing platforms. These models are based on modern methods of data analysis of object-attribute data, such as Formal Concept Analysis and biclustering. In particular, the paper is focused on the solution of two tasks – idea and antagonists recommendation – on the example of crowdsourcing platform Witology.**

***Keywords*—crowdsourcing; recommender systems; biclustering; Formal Concept Analysis**

I. Introduction

The success of modern collaborative technologies is marked by the appearance of many novel platforms for distributed brainstorming or carrying out so called "public examination". There are a lot of such crowdsourcing companies in the USA (Spigit [1], BrightIdea [2], InnoCentive [3] etc.) and Europe (Imaginatik [4]). There is also the Kaggle platform [5] which is the most beneficial for data practitioners and companies that want to select the best solutions for their data mining problems. In 2011 Russian companies launched business in that area as well. Witology [6] and Wikivote [7] are the two most representative examples of such Russian companies. Several all-Russian projects have already been finished successfully (for example, Sberbank-21, National Entrepreneurial Initiative-2012 [8] etc.). Socio-semantic networks constitute the core of such crowdsourcing systems [9], [10], [11], [12]. New approaches are needed to analyze data which underlies these networks. This paper is devoted to the new methodological base for the analysis of data generated by collaborative systems, which uses modern data mining and artificial intelligence models and methods. As a rule, while participating in a project, users of such crowdsourcing platforms discuss and solve one common problem, propose their ideas and evaluate ideas of each other as experts [13]. As a result of this process we can get a reliable ranking of ideas and users who generated them. Special means are needed in order to develop a deeper understanding of users's behavior, adequate ranking criteria and to perform complex dynamic and statistic analyses. Traditional methods of clustering, community detection and text mining should be adapted or even fully redesigned. Moreover, these methods require ingenuity for their effective and efficient use that will allow to find non-trivial results. We briefly describe models of data used in crowdsourcing projects in terms of Formal Concept Analysis (FCA) [14]. Furthermore, we present the collaborative platform data analysis system CrowDM (Crowd Data Mining), its architecture and methods underlying the key steps of data analysis.

The main part of this work is devoted to the modeling of a recommender system for crowdsourcing platforms. The way these platforms work is very different from the principles work of online-shops or specialized music/films recommender sites are based on. Crowdsourcing projects consist of several stages and results of each stage greatly depend on the results of a previous one. This is the reason why existing recommender models should be considerably adapted. The authors of this work have developed new methods for making recommendations based on well-known mathematical approaches. We propose methods of idea recommendation (for voting), like-minded persons recommendation (for collaborating) and antagonists recommendation (for contridea generation stage). It the first time the last recommendation type is proposed and it is very important for Witology platform because it features a stage of contridea generation. We have developed and tested several original methods of antagonists recommendation.

The paper consists of 6 sections. In Section II describes the platform and Witology project. Section III discusses the reasons why the company needs recommender systems. Section IV introduces basic notions of Formal Concept Analysis, describes recommender models and algorithms based on those notions. Evaluation of algorithms on real Witology data is discussed in Section V. Section VI concludes the paper.

II. Crowdsourcing platform and project Witology

One Russian proverb says "Two heads are better than one" and when dealing with crowdsourcing projects we can count on several thousands of heads. The term "crowdsourcing" is a portmanteau of "crowd" and "outsourcing", coined by Jeff Howe in 2006 [13]. There is no general definition of crowdsourcing, however it has a set of specific features. Crowdsourcing is a process, both online and offline, that includes task solving by a distributed large group of people who usually come from different organisations, and are not necessarily paid for their work.

We shortly describe the methodology of the Witology crowdsourcing company.

The clients of Witology company are banks (Sberbank), large firms (Moscow United Energy Company, RosAtom), Government institutions (Zelenograd prefecture).

The platfrom interface is simple and friendly: there are sections with the description of a project and tasks of particular stage, personal account of a user, rating tables etc. The main interface unit is a "project tree", a hierarchical structure which is growing during the project. The root of the tree is the project theme, below the root the problems are listed; each problem is connected with the related ideas how to solve the problem and so on.

There are predefined rules to maintain a project on Witology platform. The client sets up the project objectives. The project participants, crowdsourcers, undergo preliminary surveys, which include specific questions on the domain. Thus, taking into account requirements of a client and the project goal, crowdsurcers are not necessary are experts in the domain and have the experience in crowdsourcing projects. The project team is formed by a client and Witology.

There are several roles of the project team members: project manager, facilitator, manager of communities, helpdesk stuff, methodologist, content analytic, and network analytic.

The main stages of the project were "Solution's generation", "Selection of similar solutions", "Generation of counter-solutions", "Total voting", "Solution's improvements", "Solution's stock", "Final improvements" and finally "Solution's review".

After "Selection of similar solutions" by participants, analytical operations including comparison, clustering, and filtering of the ideas were performed. As a result of this stage, solutions are selected. The stage "Total voting" resulted in the selection of less number of solutions. After the stage "Solution's stock" again less number of solutions are left. From about 10-15 remaining solutions after "Solution's review" about 3-5 solutions are nominated as the best ones.

The first stage "Solution's generation" is performed individually by each user. A key difference between traditional brainstorming and the "Solution's generation" stage is that nobody can see or listen to the ideas of other participants. The main similarity is the absence of criticism which was moved to later stages. In the "Selection of similar solutions" phase participants are selecting similar ideas (solutions) and their aggregated opinions are transformed into clusters of similar ideas.

Counter-solutions generation includes criticism (pros and cons) and evaluation of the proposed ideas by means of communication between an author and experts. During this stage author of the idea can invite other experts to his team taking into account their contribution to discussion and criticism. Total voting is performed by evaluation of each proposed idea by all users in terms of their attitude and quality levels of the solution (marks are integers between -3 and 3). Two stages, i.e. "Solution's improvements" and "Final improvements", involve active collaboration of experts and authors who improve their solutions together.

The system calculates 10 ratings for each user based on their activity; these ratings include "Popularity","Social capital", "Performance", "Gamer", "Actor", "Judge", "Commenter", "Importance", "Influence", and "Reputation". For texts the company uses the following ratings: "Significance", "Influence", "Popularity", "Quality", "Attitude" and also "Reputation".

Solution's stock is one of the most interesting game stages of the project when all the participants with positive reputation rate accumulated at the previous stages receive money in internal currency "wito" and can take part in stock trade. The solutions with the highest price become winners. Finally, during the review of the solutions, experts with high reputation make their final evaluations based on several criteria in -3 to 3 scale: "Solution efficiency", "Solution originality", "Solution performance", and "Return on investment".

III. Recommender system for Witology: rationale

All the crowdsourcing projects that we analysed were very well thought out both in terms of stages and semantic content of the project. It is known that every stage is connected to a special problem which users should solve.

The completion and quality of this solution influence a lot of users ratings.

Some of these ratings coincide with the motivation system for users, others show the leadership and the winners of the project can be defined with the help of them. The authors of the work [15] have checked out the Pareto-law wich says that 20% of the users generate 80% of the content (in terms of Witology system). The results have shown that the large part of the registered users are absolutely inactive and do not participate in generation of the content. Partly it happens because of the time and effort-consuming stages. At some moment of the project most of the users understand that their every-day life and work are more important and stop taking part in the project. This leads to the situation when the project becomes a club of small amount of very active people but not the project that should have benefited from the "synergy effect" and collaborative work. This is why decreasing laboriousness of some stages is a good solution. Recommender system may help to achieve this goal.

A. Idea recommendation

Let us consider the total voting stage. All the users evaluates the ideas of each other (II). Sometimes the total number of ideas exceeds hundreds. Moreover, each idea should be evaluated according to two parameters: attitude (shows how much the user agrees with the idea) and quality (scores the idea description and validity). While evaluating the first ideas of the pool, users are really motivated and well-concentrated. At a later stage they are divided into 3 "groups": (1) continue evaluating with high quality; (2) continue doing this but only just to end the task as quickly as possible, (3) stop evaluating at all.

This work suggests to allow users evaluating not all the ideas but only the most interesting for then. Thus, we have modelled a recommender system that defines the most interesting ideas for a particular user. This algorithm is described in Section IV-B.

B. User recommendation

In addition to the idea recommendation we can recommend users to each other. The collaborative work of the "closest" users leads to more reasonable results than the work of a bitty crowd. The stage of command work is one of the stages of active user interaction. At this stage users are grouped in commands over the ideas in order to describe them better. As we have mentioned above, there is a big amount of ideas and users may have problems choosing the most interesting ones to work at. We assume that the presence of similar users in one command can help in productive discussion. Thus, the recommender system can show the other users suitable for collaborate with or the existing commands where the closest users are. All this will both help in choosing the most interesting ideas and make the command work more effective.

One more important stage of the crowdsourcing project is the stage of idea criticism (contridea generation). As we have mentioned at 2.1.4 at this stage users should criticize the ideas of each other in order to make the resulting. However, we may face the same problem: there are too many ideas, criticizing all of them is very time and effort-consuming. Sometimes people choose to criticize some ideas only because they have had a previous discussion with their authors . This situation can lead to misunderstandings and non-constructive criticism. We suggest showing only a subset of all the ideas. To choose the most interesting ideas we need an algorithm that can identify so-called antagonists: the furthest users from the target one. Then we can use this information to suggest to criticize the ideas of these antagonists. The description of the algorithm is given in SectionIV-C.

We do not address the problem of cold-start in our algorithms and recommend something only to those users who have a behavior history. However, we suggest the methodological solution of the cold-start problem for each case.

C. Evaluation of the recommender system

The recommendation quality is measured with the help of standard measures: precision and recall. The formulae:

$precision = \frac{R}{N}$

$recall = \frac{R}{(Total\_num\_of\_relevant\_items)}$

Recall for recommendations deserves a further discussion. In contrast to precision where we have a number of recommendations in the denominator, for recall we have a total number of relevant items that have almost no sense because the aim of all the above-mentioned algorithms is to decrease efforts of project participants. This means that we are planning to recommend to users doing something (group in commands, evaluate and criticize ideas) and recommend him/her only relevant tasks. So the recall will end up being low. Anyway we will observe the recall dynamics depending on different parameters of the algorithm, this should allow us to assess the adequacy of the algorithm.

It is important to say that the interpretation of precision and recall in not the same as usually in recommender systems: some data is hidden (test set) and then one compares the results of the algorithm with real values that were hidden. Firstly, all our algorithms are trained using one type of data and recommend another type of data. Secondly, the main aim of the recommender system in a crowdsourcing project is to help users and to increase the effectiveness of their work. Probably, now users group in teams not the best way and recommendations may correct that. In this case, the low precision will not necessary mean the bad result 100%. The detailed results of precision and recall estimation are presented in Section V.

IV. Recommender model and algorithms

At the initial stage of data analysis of crowdsourcing data two data types were identified: data without using keywords (links, evaluations, user actions) and data with keywords (all user-generated content). For the analysis of the 1st type of data (without keywords) we suggest to apply Social Network Analysis (SNA) methods, clustering (biclustering and triclustering [16], [17], [18], spectral clustering), FCA (concept lattices, implications, association rules) and its extensions for multimodal data, triadic, for instance [19]; recommender systems [20], [21], [22], [23] and statistical methods of data analysis [24] (the analysis of distributions and average values).

A. Formal Concept Analysis and Object-Attribute-biclustering

Methods described in this paper mainly rely on the multimodal clustering block at the analysis scheme in [25]. The protagonists of crowdsourcing projects (and corresponding collaborative platforms) are platform users (project participants) and we consider them as *objects* for analysis. More than that, each object can possess a certain set of *attributes*. User's attributes can be: topics which the user discussed, ideas which he generated or voted for, or even other users. The main instrument for analysis of such object-attribute data is FCA [14]. Let us give a formal definition. *A formal context* in FCA is a triple $\mathbb{K} = (G, M, I)$, where G is a *set of objects*, M is a *set of attributes*, and the relation $I \subseteq G \times M$ shows which object possesses which attribute. For any $A \subseteq G$ and $B \subseteq M$ one can define *Galois operators*:

$$\begin{aligned} A' &= \{m \in M \mid gIm \text{ for all } g \in A\}, \\ B' &= \{g \in G \mid gIm \text{ for all } m \in B\}. \end{aligned} \quad (1)$$

The operator $''$ (applying the operator $'$ twice) is a *closure operator*: it is idempotent ($A'''' = A''$), monotonous ($A \subseteq B$ implies $A'' \subseteq B''$) and extensive ($A \subseteq A''$). A set of objects $A \subseteq G$ such that $A'' = A$ is called a closed set. The same properties hold for closed attribute sets, i.e. subsets of the set M. A couple (A, B) such that $A \subset G$, $B \subset M$, $A' = B$ and

$B' = A$, is called a *formal concept* of a context K. The sets A and B are closed and are called *extent* and *intent* of a formal concept (A, B) respectively. For a set of objects A, a set of their common attributes A' describes a similarity of objects of the set A, and a closed set A'' is a cluster of similar objects (with the set of common attributes A'). The relation "to be more general concept" is defined as follows: $(A, B) \geq (C, D)$ iff $A \subseteq C$. By $\mathfrak{B}(G, M, I)$ we denote a set of all concepts of a formal context $\mathbb{K} = (G, M, I)$. The concepts of a formal context $\mathbb{K} = (G, M, I)$ ordered by extensions inclusion form a lattice, which is called a *concept lattice*. For its visualization a *line diagram* (Hasse diagram) can be used, i.e. the cover graph of the relation "to be a more general concept".

In the worst case (Boolean lattice), the number of concepts is equal to $2^{\{\min |G|,|M|\}}$, thus, for large contexts, FCA can be used only if the data is sparse. Moreover, one can use different ways of reducing the number of formal concepts (choosing concepts by stability [26] index or extent size). An alternative approach is a relaxation of the definition of formal concept as a maximal rectangle in an object-attribute matrix which elements belong to the incidence relation. The notion of object-attribute bicluster (OA-bicluster) [17] is one of such relaxations. If $(g, m) \in I$, then (m', g') is called *object-attribute bicluster* with the density $\rho(m', g') = |I \cap (m' \times g')|/(|m'| \cdot |g'|)$.

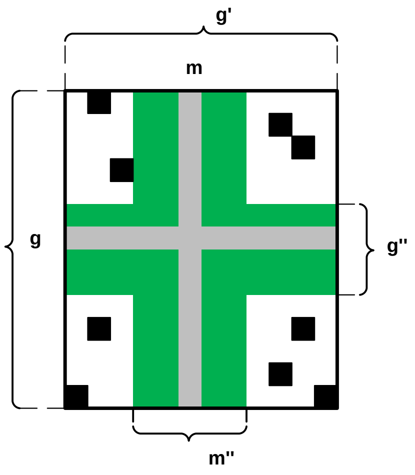


Fig. 1. OA-bicluster.

The main features of OA-biclusters are listed below:

1) For any bicluster $(A, B) \subseteq 2^G \times 2^M$ it is true that $0 \leq \rho(A, B) \leq 1$.
2) OA-bicluster (m', g') is a formal concept iff $\rho = 1$.
3) If (m', g') is a bicluster, then $(g'', g') \leq (m', m'')$.

Let $(A, B) \subseteq 2^G \times 2^M$ be a bicluster and ρ_{min} be a non-negative real number such that $0 \leq \rho_{\min} \leq 1$, then (A, B) is called *dense*, if it fits the constraint $\rho(A, B) \geq \rho_{\min}$. The above mentioned properties show that OA-biclusters differ from formal concepts since unit density is not required. Graphically it means that not all the cells of a bicluster must be filled by a cross (see fig. 1).

Besides formal lattice construction and visualization by means of Hasse diagrams one can use implications and association rules for detecting attribute dependencies in data. Then, using the obtained results, it is easy to form recommendations (for example, offering users the most interesting discussions for them). Furthermore, structural analysis can be performed and then used for finding communities. Statistical methods are helpful for frequency analysis of the different users' activities.

B. Idea recommendation – before the total voting stage

The task of this stage is to evaluate all the ideas (using marks in the range of $-3 \cdots 3$) by two parameters: attitude and quality (IV-A) Attitude shows user's opinion about the idea and indicates whether the user agrees with the proposed idea or not. Whereas the quality evaluation shows whether a user considers the proposed idea sound and well-grounded.

A problem that arises at this stage is that not all users will evaluate all the ideas, it often happens so that users start to evaluate the proposed ideas but drop the task before finishing all evaluations. There are only a few users that who evaluate all the ideas till the end.

We suggest a solution that can help to deal with the problem mentioned above: not to show all the ideas, but only the top ranked (most interesting) ones. The idea is considered to be an interesting one for a user if he/she has evaluated other posts and comments of the ideas author before.

Necessary data for the recommendation consists of matrix S (user–user) with the absolute values of evaluations sum before the total voting stage, matrix UI (idea–author): who is the author of each idea.

```
Idea recommendation algorithm
1:    for u0:
2:    [simClose, indClose]=sort(S(u0,:), desc)
3:    UNN= topN (indClose); INN=I(UNN)
4:    sort INN by simClose
5:    store Rec=topI(INN_sort)
```

As it has been mentioned, the algorithm is trained using the matrix S which describes the history of users evaluation of each others posts and comments before the total voting stage. It is important to mention that unlike the ideas evaluation (total voting) which is a special stage at a given time, posts and comments evaluation is possible during the whole time of the project. The hypothesis is that if the user 1 was interested in posts and comments of the user 2, he will be interested in his ideas as well. The rows of matrix S are the users-evaluators, the columns – the evaluated users (authors) and $S_i j$ is the sum (abs value) of all the evaluations which user i has given to all the posts and comments of the user j before the stage of total voting. The absolute value of the evaluation sum is a complex measure of the users' interest in each other: we can see the number of evaluations and their values. The high value of $S_i j$ means that the user i has given many evaluations of one sign to the user j thus he will be able to evaluate his idea adequately.

Let us look at the suggested algorithm step-by-step. Lines 23 define a neighborhood UNN by $topN$ the most interesting users for a target user and a set of potentially interesting ideas INN: the ideas of the users from UNN. After steps 2-3the ideas from INN are sorted (line 4) according to the interest of their authors in the target user (from the matrix S). Then the most interesting ideas are selected (line 5).

If the target user has not evaluated (or has not evaluated many) comments and posts of the others, the system may offer him/her to evaluate the ideas that have not yet been evaluated at all or were evaluated by just small amount of users.

C. *Antagonists recommendation for the idea criticism – before contr-idea generation stage*

The task of this stage is to criticize the ideas according to two parameters: what the idea is lacking and how it can be improved. A problem that we can encounter: the user may decide to criticize only the ideas of their rivals and this can lead to project conflicts. The way we suggest to address this problem: not to show all the ideas for criticism but only the ideas of the authors with the most opposing views (so-called antagonists). The user 1 is considered to be the antagonist for the user 2 if at the total voting stage they evaluated the ideas of others absolutely reverse. The input data for this stage will be a matrix E (user–idea) of the total voting evaluations.

1) Biclustering-based antagonists recommendation: The algorithm consists of two stages: data pre-processing (in order to find OA-biclusters) and antagonists recommendation itself. Lines 1–5 describe the first stage. As far as the algorithm receive a binary matrix from matrix E indicating whether user evaluated idea or not, it can find all the OA-biclusters (B is the set of them all). After that for any bicluster we can count so called "bicluster mass": its density multiplied by its size. According to the density definition (IV-A), a bicluster mass is just a number of non-empty cells. Then we should take top-m biclusters sorted by their mass for the future steps. The bicluster mass is a good measure in this case to sort by because it can find biclusters both with big density and/or big size. After the top-m massive biclusters are defined, the second stage of algorithm starts. Similarly to [27] the algorithm finds k nearest biclusters (lines 6–9) to the target user. The similarity between bicluster and a user is counted by the formula:

$$sim(u, b) = \frac{|I_u \cap I_b|}{|I_b|}, \text{ where}$$

I_b is a set of bicluster items and I_u is a set of users.

After calculation of k nearest biclusters (B_k) the next step, antagonists finding, starts (lines 10–18). Inside each bicluster b from B_k we count the distance between each user of the bicluster and a target user. This time we do it not with the binary matrix but with the given matrix E. The distance is counted using the Pearson correlation just by the ideas in bicluster b. Then we multiply this distance by the similarity of a target user and a bicluster b and get a matrix R (users-biclusters). An element $R(u_i, b)$ means the rating of users u_i for the target user in bicluster b. Finally, taking the mean value of this ratings by all the biclusters, we got the furthest users (by distance) from the target user and can take $top-n$ of them to recommend.

Besides the Pearson correlation we can use Hamming distance: $d = \sum_i [x_i \neq y_i]$.

In this case we give 1 for evaluations of one sign and 0 for evaluations of opposite signs. The evaluation "0" will always have different value depending on the compared evaluation (in order to be the opposite sign).

```
Biclustering-based antagonists
recommendation algorithm:
1:   B=oa-Bicluster(E_bin)
2:    for each b in B
3:        count mass(b)=dens(b)*size(b)
4:    end
5:    store top-m b by m in BM
6:    for each b in BM
7:        store sim(b,u0)
8:    end
9:    store top-k b by sim in BK
10:  for each b in BK
11:       for each u in b (except u0)
12:                 store R(ui,b)=dist(u0,ui)_b*
                                  *sim(u0,b)
13:       end
14:  end
15:  for each ui
16:  store finR(ui)=mean(R(ui,:))
17:  end
18:  store top-n ui by finR
```

2) Correlation-based antagonists recommendation: A simpler way of antagonists' recommendation is based on simple Pearson correlation (without biclusters). We just take the top-n users with the smallest correlation and recommend them. In this case we count the correlation by all the ideas (not only those which are in bicluster).

3) Hamming distance-based antagonists recommendation: One more way which we have tested is the simple Hamming distance (without biclusters data preprocessing). We should just count the Hamming distance the same way as we have described above for target user with all of other users and take $top - n$ users with the biggest distance as antagonists.

If at the total voting stage the target user has not evaluated (or evaluated too few) ideas of others, the algorithms may recommend him/her to criticize the ideas have not been evaluated till that moment.

D. *Like-minded people recommendation for teams joining*

The of the stage is to group in teams over the ideas to have an active discussion.

There is a problem: Usually users group in teams because of the interest to a particular idea and do not care about effectiveness of work with others who are in the command.

Problem solution: to show the target user the list of the closest users to group in teams with. The collaboration of likeminded people in terms of teams will do command stage more effective. The user 1 is considered to be the like-minded for the user 2 if at the total voting stage they evaluated the ideas of others the same way. Necessary data: matrix E (useridea) of total voting evaluations.

The algorithm to find like-minded people is absolutely the same as the antagonists recommendation but instead of counting users distance we count users similarity (the biggest Pearson correlation instead of the smallest). Even though we describe the methodology of like-minded persons recommendation in this work, we provide only results for the two aforementioned methods.

V. Data and experiments

For experiments we have used data of crowdsourcing project “Sberbank: No.Queues!” managed by Witology. The project took place period was from 03.09.2012 to 23.10.2012. The main topic was the problem of big queues at Sberbank offices. During the project crowdsourcers discussed the reasons why queues are appearing and the ways to get rid of them. The number of registered users was 5946 (25 of them are Witology staff). The experiment have shown that there were only about 200 more or less active users. They have generated 880 queues reasons and 1137 ideas how to solve this problem. The number of user posts (including comments) and evaluation pairs (attitude, quality) amounted to 20326 and 130000 respectively.

There were following stages at the project:

1) Problem determination;
2) Total voting for problems;
3) Idea generation;
4) Selection of similar ideas;
5) Idea verification;
6) Team work at ideas;
7) Contr-idea generation;
8) Total voting for ideas;
9) Command work at top-50 ideas;
10) Top-50 ideas review.

The recommendation algorithms were performed using Matlab. An additional supporting program, that finds oa-biclusters was written in Python. As a result of data pre-processing we have found 19,405 biclusters (without density threshold). After that we have used 20 largest biclusters ($m = 20$).

A. Idea recommendation

For the idea recommendation algorithm we have used a matrix of the user-user type (rows are for the evaluators and columns are for the evaluated users), where each cell shows absolute value of all the evaluations of the respective evaluator-evaluated user pair before the idea voting stage. Generally there were 117 evaluators and 139 evaluated users. The assumption is that if the user 1 was interested in some posts and comments of user 2 before the total voting stage for ideas, his ideas will be interesting for the user 1 as well (at the total voting stage). In order to evaluate the algorithm weve used the matrix user-idea which reflects the real situation of idea evaluation (from the total voting stage for ideas). We have looked at classic measures: precision and recall. Two parameters were being changed: $topN$ – the number of the closest neighbors and $topI$ – the number of recommended ideas.

We will variate these parameters one by one and fix others.

1) The precision and recall dependence of neighborhood size: The number of recommendations is fixed: $topI = 20$

So, the recall is really not high as we have expected. The precision stops growing at about 0.65 value. There are two possible explanations. Firstly, our hypothesis may be wrong and users evaluate ideas regardless their previous evaluations. Secondly, the total voting stage is really very hardworking and time consuming. Moreover, this stage is obligatory and each user should evaluate all the ideas. Often users do it not very

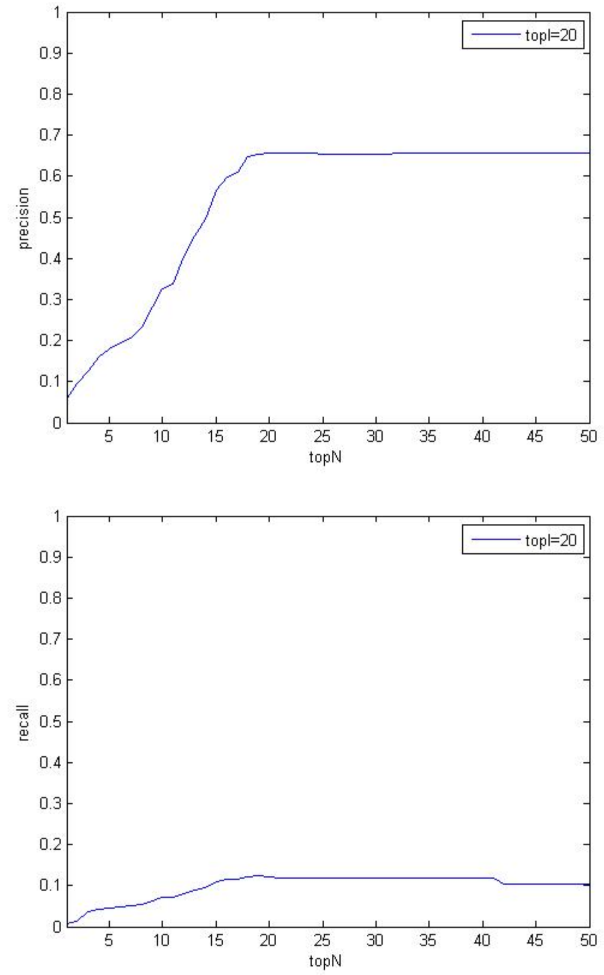


Fig. 2. The precision (a) and recall (b) dependence of neighborhood size

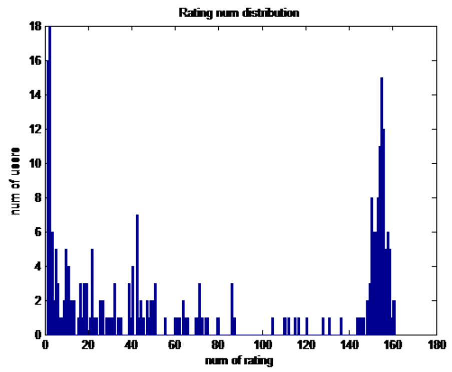
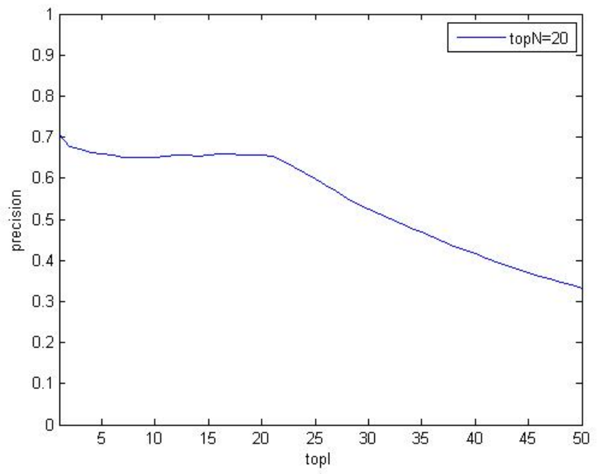


Fig. 3. The distribution of evaluations number at total voting stage.

thoroughly because of a big work load. Let us look at the distribution of evaluations numbers in Fig. 3(at total voting stage):

In fact, a great number of users either evaluates almost all the ideas (172 in our case) or just less than 10 ideas. The moment when the precision and recall stops changing (about 20 users in neighborhood) may mean that the algorithm reach



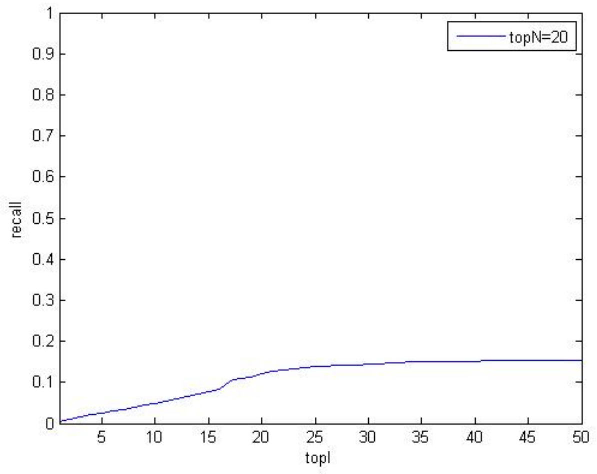


Fig. 4. The precision (a) and recall (b) dependence of number of recommendations.

the necessary number of user neighborhood to give relevant recommendations.

2) The precision and recall dependence on recommendations number: The number of recommendations is fixed: $topN = 20$

Firstly, let us point out a high level of precision. It decreases while the recommendation number grows. It is explained quite well: the more ideas we recommend the less is the part of guessed correctly and the more is the probability to make correct recommendations for particular user (recall increasing). As it was foreseen, the recall is quite low. But the tendency of its growth let us assume that the algorithm works correctly.

B. Antagonists recommendation for the idea criticism

For antagonists recommendation a matrix of idea evaluations at total voting stage was used (user-idea). In order to evaluate recommendations quality we have generated a binary matrix user-user where 1 means that user 1 has participated in users 2 idea criticism, 0 – otherwise. Finally we got a matrix 62×73 because some users havent participated at this stage

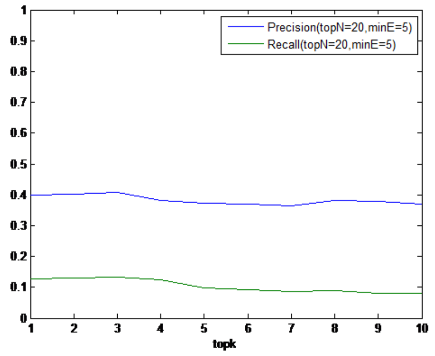


Fig. 5. Precision and recall dependence of the closest biclusters number

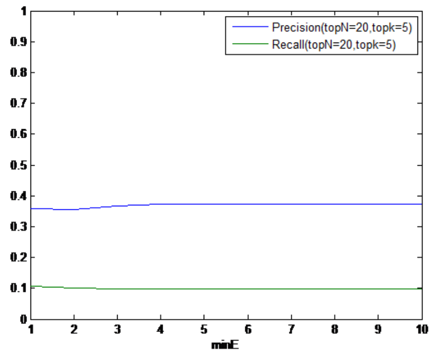


Fig. 6. Precision and recall dependence on the minimal required number of evaluations

at all. Quality measures are the same: precision and recall. Algorithm parameters are the following:

- $minE$ – the minimum required number of evaluated ideas for giving recommendation to a person
- $topk$ – the number of the closest biclusters in the target users neighborhood
- $topN$ – the number of recommended items.

The average number of ideas that user criticized is 20 that is why when $topN$ is fixed it equals to 20.

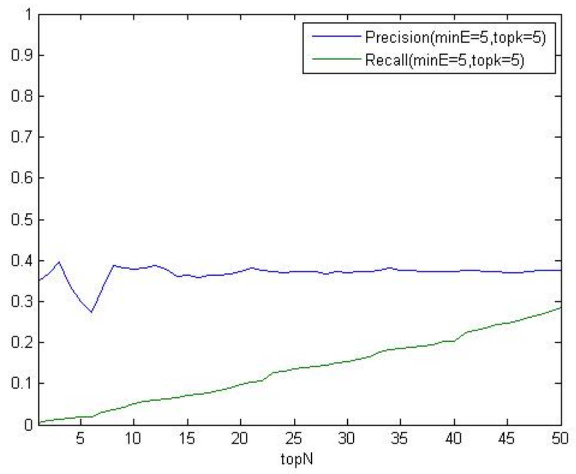
1) Precision and recall dependence of the closest biclusters number: Fixed algorithm parameters $minE = 5$ and $topN = 20$ give us the following results Fig. 5.

The precision is quite good. The decreasing tendency is observed: that may happen because of some noise presence in biclusters neighborhood. We do not show the results of like-minded people recommendations in this paper, but we should note that precision here is much bigger. For new methodology of recommendations (antagonist), it is a very good indicator.

2) Precision and recall dependence on minimal required number of evaluations: Fixed algorithm parameters: $pk = 5$, $topN = 20$

The same result is for like-minded people recommendations: this parameter is not meaningful. Precision and recall dependence of recommendations number

Fixed algorithm parameters: $topk = 5$, $minE = 5$



Precision and recall dependence of recommendations number

The precision is almost the same but recall increases quite logically.

3) Comparison with other distance measures: The most interesting dynamic result is observed depending on the recommendations number. Changing this parameter, we will compare the quality of this antagonists recommendation algorithm (based in biclusters) with others: based on only Pearson correlation (CosineBased) and based on only Hamming distance (SimpleHamming). Moreover, the similarity (distance) in biclusters-based algorithm is calculated using two ways: with a help of Pearson correlation (Bicluster, as it was described before) and with a help of Hamming distance (BiclusterHamming).

Figures 7 show that when the recommendations number is low the best result has BiclusterHamming and when the recommendations number is high two algorithm are good: Bicluster and SimpleHamming. The initial aim was to shorten user effort and time, so we are interested in low recommendations number. In this case BiclusterHamming is the real winner: the precision is almost 0.5. Now let us compare algorithms quality and their work time.

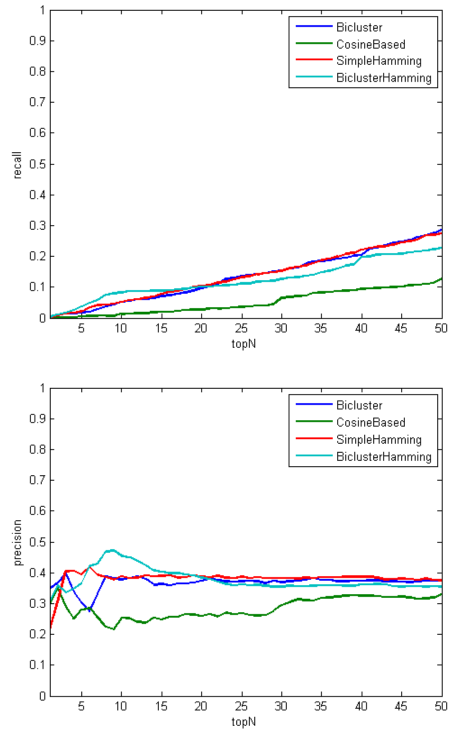


Fig. 7. Different antagonists recommendation algorithm precision (a) and recall (b)

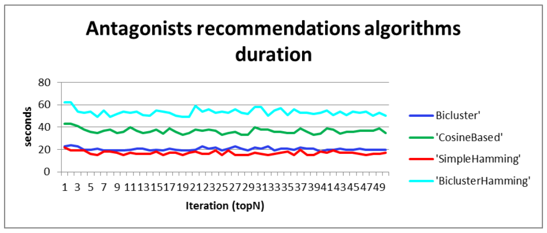


Fig. 8. Antagonists recommendations algorithms duration

Despite the high precision, BiclusterHamming has the worst time results. However, when the recommendations number is low (our case) it may be not so critical. Let us also point out that in comparison with information retrieval algorithms where CosineBased is one of the best, here it is both slower and worse in terms of quality than SimpleHamming.

VI. Conclusion

This work describes the mathematical model for crowdsourcing projects and the Recommendation system, both based on Formal Concept Analysis and oa-biclustering. The recommendations of ideas and users-antagonists were modelled. The last type of recommendations is a methodologically novel one. As for antagonists recommendations, we have tested several algorithms: with data pre-processing (oa-biclustering) and without. The results show that using biclusters improves recommender system quality. More than that, two ways of antagonists determination were tested: using negative Pearson correlation and big Hamming distance. When the recommendations number is low the best is BiclusterHamming (based on biclustering and using Hamming distance for antagonists determination). As for the idea recommendations, the precision of this algorithm is much higher than the same one used for antagonists recommendations. The algorithm's originality is that users' similarity is calculated not based on recommended ideas but other types of objects (posts and comments). Experiments and observations at a real crowdsourcing projects let us conclude that recommender system implementation should be started from idea recommendation at a total voting stage. Firstly, the precision is higher, secondly this stage is very time consuming and sometimes boring, therefore decreasing a number of ideas for evaluation should make crowdsourcing projects more attractive for participants. A cold start problem is not solved in this work – in case of previous users inactivity he/she gets no recommendations. A solution for a cold start problem will be addressed in the future work, we describe one possible solution in this work methodologically.

Acknowledgment

The study was conducted in the framework of the Basic Research Program at the National Research University Higher School of Economics (HSE) in 2013 and 2014 and in the Laboratory of Intelligent Systems and Structural Analysis at HSE as a part of the project "Mathematical Models, Algorithms, and Software Tools for Intelligent Analysis of Structural and Textual Data". Second author was also supported by Russian Foundation for Basic Research (grant #13-07-00504).

References

[1] Spigit company. [Online]. Available: http://spigit.com/

[2] Brightidea company. [Online]. Available: www.brightidea.com/

[3] Innocentive comp. [Online]. Available: http://www.innocentive.com/

[4] Imaginatik company. [Online]. Available: http://www.imaginatik.com/

[5] Kaggle. [Online]. Available: http://www.kaggle.com

[6] Witology company. [Online]. Available: http://witology.com/

[7] Wikivote company. [Online]. Available: http://www.wikivote.ru/

[8] Sberbank-21, national entrepreneurial initiative-2012. [Online]. Available: http://sberbank21.ru/

[9] C. Roth, "Generalized preferential attachment: Towards realistic socio-semantic network models," in *ISWC 4th Intl Semantic Web Conference, Workshop on Semantic Network Analysis, Galway, Ireland,*, ser. CEUR-WS Series (ISSN 1613-0073), vol. 171, 2005, pp. 29–42.

[10] J.-P. Cointet and C. Roth, "Socio-semantic dynamics in a blog network," in *CSE (4)*. IEEE Computer Society, 2009, pp. 114–121.

[11] C. Roth and J.-P. Cointet, "Social and semantic coevolution in knowledge networks," *Social Networks*, vol. 32, pp. 16–29, 2010.

[12] R. Yavorsky, "Research Challenges of Dynamic Socio-Semantic Networks," in *CEUR Workshop proceedings Vol-757, CDUD'11 - Concept Discovery in Unstructured Data*, D. Ignatov, J. Poelmans, and S. Kuznetsov, Eds., 2011, pp. 119–122.

[13] J. Howe, "The rise of crowdsourcing," *Wired*, 2006.

[14] B. Ganter and R. Wille, *Formal Concept Analysis: Mathematical Foundations*, 1st ed. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1999.

[15] A. Bezzubtseva and D. I. Ignatov, "A New Typology of Collaboration Platform Users," in *CEUR Workshop proceedings Vol-757, EEML'12 - Experimental Economics and Machine Learning*, R. Tagiew, D. I. Ignatov, A. A. Neznanov, and J. Poelmans, Eds., 2012, pp. 9–19.

[16] S. Barkow, S. Bleuler, A. Prelic, P. Zimmermann, and E. Zitzler, "Bicat: a biclustering analysis toolbox," *Bioinformatics*, vol. 22, no. 10, pp. 1282–1283, 2006.

[17] D. I. Ignatov, A. Y. Kaminskaya, S. Kuznetsov, and R. A. Magizov, "Method of Biclusterzation Based on Object and Attribute Closures," in *Proc. of 8-th international Conference on Intellectualization of Information Processing (IIP 2011). Cyprus, Paphos, October 17–24.* MAKS Press, 2010, pp. 140–143, (in Russian).

[18] D. I. Ignatov, S. O. Kuznetsov, R. A. Magizov, and L. E. Zhukov, "From Triconcepts to Triclusters," in *Proceedings of the 13th international conference on Rough sets, fuzzy sets, data mining and granular computing*, ser. RSFDGrC'11. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 257–264.

[19] R. Jäschke, A. Hotho, C. Schmitz, B. Ganter, and G. Stumme, "TRIAS–An Algorithm for Mining Iceberg Tri-Lattices," in *Proceedings of the Sixth International Conference on Data Mining*, ser. ICDM '06. Washington, DC, USA: IEEE Computer Society, 2006, pp. 907–911.

[20] D. I. Ignatov and S. O. Kuznetsov, "Concept-based Recommendations for Internet Advertisement," in *Proc. CLA 2008*, ser. CEUR WS, R. Belohlavek and S. O. Kuznetsov, Eds., vol. Vol. 433. Palacky University, Olomouc, 2008, 2008, pp. 157–166.

[21] D. Ignatov, J. Poelmans, and V. Zaharchuk, "Recommender System Based on Algorithm of Bicluster Analysis RecBi," in *CEUR Workshop proceedings Vol-757, CDUD'11 - Concept Discovery in Unstructured Data*, D. Ignatov, J. Poelmans, and S. Kuznetsov, Eds., 2011, pp. pp. 122–126.

[22] D. I. Ignatov, J. Poelmans, G. Dedene, and S. Viaene, "A New Cross-Validation Technique to Evaluate Quality of Recommender Systems," in *PerMIn*, ser. LNCS, M. K. Kundu, S. Mitra, D. Mazumdar, and S. K. Pal, Eds., vol. 7143. Springer, 2012, pp. 195–202.

[23] D. I. Ignatov, A. V. Konstantinov, S. I. Nikolenko, J. Poelmans, and V. Zaharchuk, "Online recommender system for radio station hosting," in *BIR*, ser. Lecture Notes in Business Information Processing, N. Aseeva, E. Babkin, and O. Kozyrev, Eds., vol. 128. Springer, 2012, pp. 1–12.

[24] A. Clauset, C. R. Shalizi, and M. E. J. Newman, "Power-law distributions in empirical data," *SIAM Rev.*, vol. 51, no. 4, pp. 661–703, Nov. 2009.

[25] D. I. Ignatov, A. Y. Kaminskaya, A. A. Bezzubtseva, A. V. Konstantinov, and J. Poelmans, "Fca-based models and a prototype data analysis system for crowdsourcing platforms," in *ICCS*, ser. Lecture Notes in Computer Science, H. D. Pfeiffer, D. I. Ignatov, J. Poelmans, and N. Gadiraju, Eds., vol. 7735. Springer, 2013, pp. 173–192.

[26] S. O. Kuznetsov, "On stability of a formal concept," *Ann. Math. Artif. Intell.*, vol. 49, no. 1-4, pp. 101–115, 2007.

[27] P. Symeonidis, A. Nanopoulos, A. N. Papadopoulos, and Y. Manolopoulos, "Nearest-biclusters collaborative filtering based on constant and coherent values," *Inf. Retr.*, vol. 11, no. 1, pp. 51–75, 2008.