Soft Computing (2020) 24:11121–11134
https://doi.org/10.1007/s00500-020-04696-z

K-OpenAnswer: a simulation environment to analyze the dynamics of massive open online courses in smart cities

Filippo Sciarrone[1] · Marco Temperini[2]

Published online: 22 January 2020

Abstract
The smartness of a city is given by the technologies it put to use, and more than that, by the people empowered by such technologies; it is worth thinking about how people can be trained to be empowered by smart technologies, and how cities can become "educational." So, while sustainability and technology solutions for smart cities are strategic challenges, one of these is surely distance education and training. In this field, the Web offers many opportunities, such as the e-learning platforms where students can learn, according to their own needs and pace. The massive open online courses (MOOCs) are particular distance learning platforms, generally offering, so far, free courses on a huge amount of topics, and characterized by a (potentially) very high number of enrollments. In a MOOC, a teacher, or tutor, has a hard life when trying to follow and manage with the learning processes of thousands of students. In particular, assessment can be managed almost exclusively by letting the student answer questions in closed answers tests. This strategy has some didactic limits, while a valid alternative is to use peer assessment (PA) over more articulated assessment activities (e.g., open-ended questions). PA makes students grade their peers' answers, and provides learners with significant advantages, such as refining their knowledge of the subject matter, and developing their meta-cognitive skills. In this work, we present a software platform called K-OpenAnswer, which helps teachers to simulate the dynamic of a MOOC where PA is used. The system uses a machine learning technique, based on a modified version of the K-NN algorithm, and provides teachers with a statistical environment by which they can monitor the evolving dynamic of a simulated MOOC, according to the techniques we use to implement PA. An experimental evaluation is presented that highlights the advantages of using the system as a valid tool for the study of real MOOCs.

Keywords Smart cities · Peer assessment · Machine learning · E-learning

1 Introduction

In the debate about smart cities and the change of pace, they can support in social evolution, important issues regarding the awareness and competence of the smart citizens (Lytras and Visvizi 2018), the perception such citizens have of the smart services, as well in wide urban ecosystem as in smaller living environments (Visvizi and Lytras 2018), and the role of social networks in spreading the opportunities offered by the smart services (Lytras et al. 2018).

One aspect of the "smart" concept for cities, regions, social connection and social transformation is in a people-centered approach. It involves many areas, in which social networking, continuing education and social learning are comprised.

In this paper, we deal with educational techniques, based on peer evaluation and peer learning, instantiated on the case of *massive open online courses* (MOOCs), where the class can more resemble a social learning ecosystem. The technique we propose has been developed with the purpose of making it applicable, in real time, to a huge number of learners, so susceptible of applications in extended social learning environments, to promote *just in time* and *lifelong* learning. We use methods to model the user that involves a cross-analysis of their representations in the system, based on their dependencies, and on the information flowing among them. The user can be the learner, in our proposal, or the citizen,

Communicated by Miltiadis D. Lytras.

✉ Filippo Sciarrone
sciarro@ing.uniroma3.it

Marco Temperini
marte@diag.uniroma1.it

[1] Department of Engineering, ROMA TRE University, Via della Vasca Navale, 79, Rome, Italy

[2] DIAG-Department of Computer, Control, and Management Engineering, Via Ariosto, 25, Rome, Italy

in a lifelong learning prospective. The whole set of users' (learners') models is constantly updated during the process, until the models reach a satisfactory stability, at least locally to the current learning aims. These techniques use soft computation methodologies, such as machine learning, to allow for a better assessment of learning tasks, in general, and for a better mutual assessment among the learners.

Such a learning ecosystem can be nurtured through the learners' participation together with teacher's coordination, so to display the advantages of a (learning) social network, certainly a focused one (on a subject matter for instance) but available to extend, and serve wider areas of competence in a more general purpose social network.

1.1 Education in the digital era

In recent years, education has been adapting to the effects of the digital revolution, due to the exponential growth both of the Internet and of multimedia devices. Such two components have undoubtedly empowered the global educational offer, for instance with the possibility to propose to highly interactive and engaging educational materials. The exponential growth of the Internet has increased distance education through the use of the e-learning platforms, web-accessible, via any connected device, 24 hours a day. They have been providing education for years to universities, schools, communities of practice and in general to all areas of education. *Moodle*,[1] *Docebo*[2] and *Sakai Project*[3] are widely known examples of such platforms. Furthermore, over the past two years, *Social* platforms have also been emerging, such as *Edmodo*, *Fidenia* and *weSchool*. Social platforms are based on the *Social Learning* paradigm where the learning process takes place also by peers' interaction. Other types of platforms that are becoming increasingly popular today are those supporting MOOCs. Basically, MOOCs are online e-learning platforms, where learners can enroll and take courses out of a huge variety of topics, so far mostly free of charge. To date, over 900 universities around the world have launched at least one MOOC. MOOC providers are also partnering with companies to launch courses, mostly on technology-related topics. The numbers of students enrolled to MOOCs was in 2015 about 58 million; in 2018, more than 11 thousands MOOCs were offered or announced, growing from 9400 and 6850 of the previous years (https://www.classcentral.com/report/mooc-stats-2018/). *Coursera*[4] is one of the most prominent platforms, and the largest to date, with at least 1700 active courses and over 23 million of enrolled students, in courses on economics, computer science, mathematics, logic, engineering, personal development, languages and communication. In MOOCs, the number of students means richness, from both a cultural and economic point of view, while it poses serious challenges to the teachers/tutors/instructors, as they have a little possibility to personally follow the learning processes of each student. So it is important to adopt special learning strategies, capable to help students in their learning, and the teacher in monitoring students' learning paths. Assessment is a crucial aspect in learning strategies; aside of taking the learning material, the learner has to undertake complex tasks, such as answering questions, learning from the assessment of such answers and in many cases assessing others' answers. All such activities are crucial, in order to help students to train their meta-cognitive skills (Metcalfe and Shimamura 1994), and their knowledge on the subject matter, to let it grow at higher cognitive levels (Li et al. 2010; Bloom et al. 1956; Anderson and Krathwohl 2000).

Peer assessment (PA) is an educational strategy that helps achieving such enhancements (Sadler and Good 2006). In a PA session, students can be exposed to open-ended questions, a tool widely used when one wants not to go to the extremes of either multiple-choice quizzes or full-fledged essays. Open-ended questions can implement a wide variety of tasks, such as (i) short essays (presenting a solution and its justifications, possibly using complex formulae); (ii) programming code (with attached technical description); and (iii) free text answers (proposing solutions to a less tightly specified problem). Open-ended questions result in a more challenging and informative source of analysis both for students and teachers than multiple-choice tests (Palmer and Richardson 2003). One aspect of PA is that in the educational practice, it can be complemented and possibly decisively enriched by the teacher's intervention, namely the answers' grading. To provide the learner with help and awareness, the teacher's assessment and PA of the answers could be very important. In fact, a learner will certainly learn from teacher's evaluation of her/his answer, but (s)he can learn much from PA as well (Sadler and Good 2006) (especially from good PA). On the other hand, grading is a time-expensive activity for the teacher, and doing it for all the students in a MOOC class would be prohibitive, with the result of not having PA at all, or having it in a crippled fashion. There is the option, though, of having the teacher to grade only a subset of the answers, using such grading to *tune*, or *calibrate*, or *mediate* PA and get anyway to a good feedback for each learner (Piech et al. 2013; Sterbini and Temperini 2013; Suen 2014; Reynolds and Moskovitz 2008).

1.2 Content of the paper and research questions

In this paper, we present the *K-OpenAnswer* software platform, a standalone computer system designed to simulate

[1] https://moodle.org/.

[2] https://www.docebo.com/.

[3] https://www.sakailms.org/.

[4] https://www.coursera.org/.

a MOOC. We assume that in a PA session, each peer has answered an open-ended question, and peer-assessed some of her/his peers' answers. Then, the final grading of the answers is based on the PA, mediated by the teacher's grading work, performed on a subset, as minimal as possible, of the answers. Based on the PA data, and on the information added into the system by the teacher's grading of some answers, the remaining answers are given automated grading.

In the K-OpenAnswer approach, the PA provides an initial student model (SM), computed based only on the peers' evaluations. The SM is then enhanced through progressive updates, determined by the use of the grades that the teacher will add into the system by grading some of the peers answers. Eventually, the updated SMs are used to infer the remaining grades.

Basically, the SM is a representation of the individual learner's skills on the topic of the question (represented by the K variable) and of her/his capability to assess correctly the peers' answers (represented by the J variable). In the previous works, we have presented the *OpenAnswer* framework operating along the lines above described, grounded on a student modeling process based on a Bayesian Network (BN) dynamic (Sterbini and Temperini 2012b; De Marsico et al. 2017a). The use of BN in student modeling can be effective when the number of peers is not too high. In the case of a MOOC, such an approach is made expensive by the computational complexity of the algorithms involved in its management (and by the sheer amount of data to manage).

To overcome such difficulties, we have previously introduced a different approach, based on the use of a modified version of the K-NN machine learning method (Mitchell 1997; De Marsico et al. 2017b).

K-OpenAnswer is the implementation of the above-mentioned approach. Here, we present this implementation and show how it is possible to use it both for administering a PA session, and for simulating one. The possibility of simulating a session is of particular importance in our research area, as it allows to test the implementation of the framework without enduring a full-fledged real-world experimentation.

Through simulation we obtained an evaluation of the system that we present in this paper.

The evaluation shows how the system can simulate MOOCs dynamics, once the class composition is initially stated (in terms of number of students and distribution of model values). A short description of the system simulation activities is as follows:

- Initially a class is created, defining the number of students and configuring each SM by means of a Gaussian distribution, over the values of K, the knowledge level and J, the assessment capability, as explained in De Marsico et al. (2018). According to these distributions, the PA is computed (i.e., the simulated learners' PAs);
- At this stage, the class is in place, and the PA has been held, so the teacher can start grading. Each student is represented in the (K, J) space, as a point whose coordinates are the model values. So the teacher can have a graphical representation of the class, and appreciate how the students are spread. Basing on this representation, the teacher can choose what answer to grade next. When an answer is graded, a mechanism of propagation of this new information is executed. The models of all the students related to the new grade (the answer's author, and its peer graders) are updated. Accordingly, all the students are reclassified, i.e., their models are re-computed, changing their positions in the (K, J) space;
- By repeating the grading operation, the teacher can see the class representation evolving in the (K, J) space, and appreciate it. Let us remember that for a student, K represents the correctness of the answer, while J the correctness of the assessments (s)he gave. So, for instance, the teacher could see how the general models are getting better or worse, or how the proposed exercise was more or less tough for the current state of the learners.

We will base our evaluation on two research questions (RQs):

RQ1: Does the K-OpenAnswer System allow the teacher to simulate a MOOC, and to appreciate its evolution?

Here, we intend to consider the evolution of the MOOC during the grading phase of a single PA session. Of course at the end of each PA session, the system provides a picture, by the distribution of the SMs in the space. By comparing these pictures along several PA sessions held during the semester, the teacher could appreciate the evolution of the class in time.

RQ2: Is the analysis supported by K-OpenAnswer conducive to didactic suggestions for the teacher?
Here, we point at some possible hints that a teacher can receive, about the students learning process, and the exercises effectiveness, while visualizing the MOOC's evolution.

1.3 Structure of the paper

The structure of the paper is as follows: Section 2 depicts some related work. Section 3 shows the SM which is behind each student and on which the network learning process is based. In Sect. 4, the K-NN rules are illustrated, while in Sect. 5, we present the K-OpenAnswer system and discuss some of its main characteristics. Section 6 depicts the evaluation of the system and the last section provides some conclusions and discussion of future work.

2 Related work

PA (Kane and Lawler 1978) is an activity where a student (or a group of students) is allowed to evaluate other students' assignments. This activity can be organized in different ways, yet a basic aspect is that it can be considered as one of the tasks by which social interactions and collaborations among students can be triggered. It can also serve as a way to verify how the teacher can communicate to the students on her own quality requirements with respect to the learning topics; if this happens, assessments from peers and from teacher agree better (Sadler and Good 2006). Moreover, there are empirical evidences of the benefits of using PA approaches. In Tenório et al. (2016), the authors show that about 60% of the studies reveal improvements in student performance, after PA activities, and about $\frac{1}{3}$ of the studies states that PA is beneficial for teachers as well.

In relation to MOOCs, the ideal peer grading system should (1) provide highly reliable/accurate assessment, (2) allocate a balanced and limited workload across students and course staff, (3) be scalable to class sizes of tens or hundreds or thousands of students and (4) apply broadly to a diverse collection of problem settings (Piech et al. 2013). However, in most MOOCs, due to big numbers, and to the consequent lack of individual support to students, discussion forums remain the only channel to exchange information and clarifications among peers. On the other hand, several studies point out the limitations of discussion forums, such as low overall participation (Kizilcec et al. 2014; Onah et al. 2014) and possible lack of responsiveness (Yang et al. 2014). Consequently, there is a gap between the goal of supporting students by establishing a learning community, and the current practical implementations of collaboration mechanisms. It has been argued that effective ways of supporting collaborative learning in MOOCs, also taking into account the problems given by the asynchronous communication and by the heterogeneity of the population, are still to be found (Penstein Rosé and Ferschke 2016). In this respect, personalization, support in finding peers to have information exchange, and support to the formation of learning groups, have been considered.

Moreover, the learning activities of a class in a MOOC generate quite large amounts of data. (The data coming from sessions of PA are an example). Hence, several contributions in literature point out the usefulness of Machine Learning, and more in general artificial intelligence techniques, to visualize and analyze the dynamics of the class, as both a group of individuals and a community of learners (Limongelli et al. 2008, 2013, 2015).

In De Marsico et al. (2017a), the *OpenAnswer* system is presented, as managing PA over open-ended questions, based on the representation of learners models and peer assessments by Bayesian networks. The learner's SM is defined by stochastic variables (K representing the learner's *Knowledge* about the question's matter, and J holding the learner's ability to "judge" a peer's answer). Other variables represent the correctness of an answer, and the learners' peer evaluations. Conditional probability tables (CPT) model the dependencies among the variables. The *OpenAnswer* workflow needs that each peer mark n answers of other peers. Then, the teacher starts grading, and, with each new grade, adds information in the BN. The information propagates and the variables are updated consequently. The grading process continue, until a termination condition is met; then, all the answers that were not graded by the teacher are assigned a grade computed based on the current (final) value of correctness.

OpenAnswer is quite configurable; there are several algorithms available to suggest the teacher what would be the best next answer to grade, and to compute the final grade. Also the CPTs nested in the system can be learned out of the class data from previous PA sessions, allowing to adapt the system to the class and its teacher(s). However, the BN is basically a black box, and nothing of it is shown to the teacher; so the feedback for the teacher is limited to the grades and models produced by the process. Moreover, the use of BN brings in computational problems that make it difficult to use the system when too many peers have to be dealt with (which is precisely our case when a MOOC is implicated). The system we are presenting in this paper try to solve the above problems, applying a different machine learning approach, computationally lighter and entailing the possibility to provide the teacher with simple graphic representations of the class, to allow analyzing the class dynamics.

Another work (Anson and Goodman 2014) proposes peer assessment to improve student team experiences. An online peer assessment system and team improvement process were developed based on three design criteria, namely efficient administration of the assessment, promotion of quality feedback and fostering effective team processes.

In Sterbini and Temperini (2012a), the authors propose an approach to open answers grading, based on constraint logic programming (CLP) and peer assessment, where students are modeled as triples of finite domain variables. The CLP Prolog module supported the generation of hypotheses of correctness for answers (grounded on students peer evaluation), and the assessment of such hypotheses (also based on the answers already graded by the teacher).

In Piech et al. (2013), the authors formulate and evaluate intuitive probabilistic peer grading models for estimating submission grades and grader biases/reliability. This could allow for compensating grader idiosyncrasies. As in our work, the authors address peer grading as a critical tool for scaling the grading of complex, open-ended assignments, to accommodate courses with tens or hundreds of thousands of students. They propose some algorithms for estimating and correcting grader biases and reliability, showing consequent improvements in peer grading accuracy. They use real data, based on 63,199 peer grades given in a course in Human Computer Interaction offered by Coursera. The probabilistic approach does not take into account a student model. This is because the experimentation was done on real data.

3 The student model

In this section, we illustrate the SM, which is behind the MOOC's learning processes. This model is the result of an evolution, of which different development steps have been reported in the previous papers (De Marsico et al. 2017b, 2018). Each student is represented by a set of variables, $SM \equiv \{K, J, Dev, St\}$, where $K \equiv [1, 10]$ is the grade of the learner's answer in the PA session. It represents the learner's competence (*Knowledge*). The variable $J \equiv [0, 1]$ is a measure of the learner's assessing capability (*Judgement*). It depends on K, under the assumption that to grade effectively an answer, knowing about the matter of the question is necessary. The standard deviation Dev represents the credibility of the current value of K; it is computed on the grades given by peers to the student answer in the session; the higher this value, the less K is given assured credit. Finally, $St \equiv \{CORE, NO\_CORE\}$ represents the *student state*. Each student can be in two different states: *CORE* (the student's answer has been graded by the teacher) and *NO_CORE* (so, only peers graded the answer so far). Initially, all the students are *NO_CORE*. A *NO_CORE* student is represented as s^-. *CORE* students are represented by s^+, and their K is the teacher's grade. The community of students is, at any given moment, dynamically parted into two groups: the *Core Group* (CG) of the *CORE* students (we shall also use S^+), and its complement $\overline{CG}$ (also S^-). By this representation, each learner can be represented as a point in the two-dimensional space (K, J).

Figure 1 represents the distribution of the (K, J) values in the (K, J) space. This kind of students representation allows for a clear monitoring of the learning process.

3.1 Student model initialization

Each SM is initialized by means of a PA. This phase evolves as follows:

1. The tutor assigns an open-ended question to all the MOOC students;
2. Each student grades the answers of n different peers, while she receives n peer grades;
3. Each SM is initialized as follows:

$$K_l^- = \frac{\sum_{i=1}^{n} K_i^-}{n} \tag{1}$$

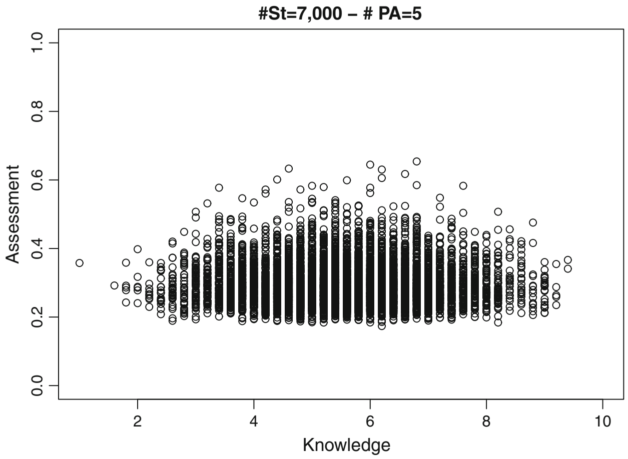


Fig. 1 An example of a MOOC composed by 7000 learners and five PAs

where K_i^- is the grade received by the ith of the n peers who graded the s_l^- student;
4. For each s_l^- student, J_l^- is initialized as follows:

$$J_l^- = \frac{1}{1 + \sqrt{\sum_{i=1}^{n} \Delta_i{}^2}} \tag{2}$$

$\Delta_i^2 = (K_{lj} - \overline{K_j})^2$, being K_{lj} the grade assigned by the student s_l to the student s_j and $\overline{K_j}$ the arithmetic mean, i.e., the initial K^- of the student s_j, computed by Eq. 1;
5. The standard deviation Dev is computed:

$$Dev_i = \sqrt{\frac{\sum_{l=1}^{n} (K_i - K_l)^2}{n}} \tag{3}$$

where K_l is the grade of the peer l, belonging to the set of the n students that graded her in the PA step. Otherwise, this value is the teacher's grade.
6. All students are initialized to $St = NO\_CORE$.

At each step, the module changes the SMs in S^-, until a termination condition suggests to stop cycling, i.e., the inferred S^- students' grades distribution is sufficiently close to the teacher's distribution. Figure 1, shows a community of $n = 7000$ peers, built by means of a PA of $five$ grades for each learner. Figure 2 shows the peers grades distribution of the same community, while Fig. 3 depicts an example of the Dev distribution after the initial PA session.

3.2 Teacher grading

After the SM initialization, all learners belong to the S^- set. Each learner evolves in the (K, J) space as follows:

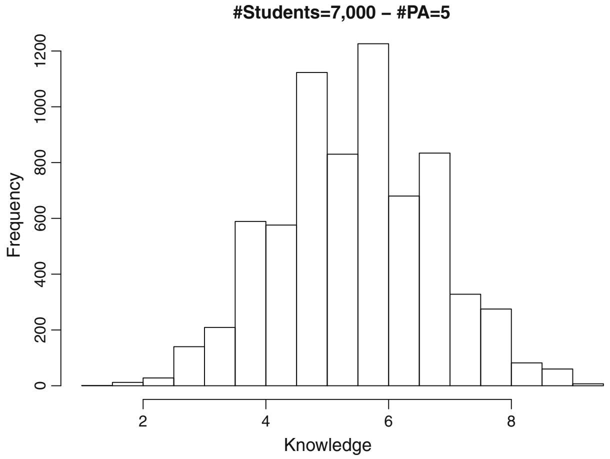


Fig. 2 PA distribution for the example of Fig. 4, concerning the K distribution

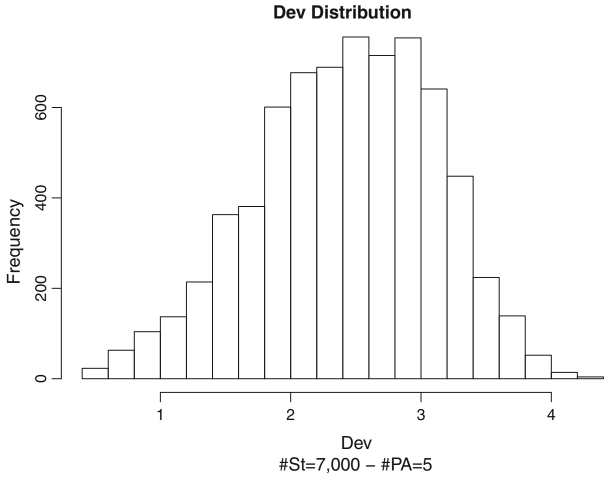


Fig. 3 PA distribution for the example of Fig. 4 concerning the Dev distribution

1. The teacher is suggested a ranked list of students/answers to grade, sorted by the Dev key; this choice represents the possibility for the teacher to grade the least reliable values among the votes of each student;
2. The teacher selects a group of students in the ranked list, and grades their answers. Such grades are the new, final, K^+ values for such students;
3. The graded students become s^+ students, and their position in the (K, J) space changes. First, the K value:

$$K^+ = K_{\text{teacher}} \tag{4}$$

and after, the J value:

$$\begin{aligned} J^+_{\text{new}} &= J_{\text{old}} + \alpha(JMAX - J_{\text{old}}) \quad (0 \le \alpha \le 1) \\ J^+_{\text{new}} &= J_{\text{old}} + \alpha J_{\text{old}} \;\; \text{with} \;\; (\alpha < 0) \\ \alpha &= \frac{K_{\text{teacher}} - K^-_{\text{old}}}{IMAX} \end{aligned} \tag{5}$$

Here and in the following, we use $KMIN$ and $KMAX$ to denote the minimum and maximum values for K (i.e., here respectively 1 and 10). $IMAX$ will denote the maximum difference between two values of K, i.e., here 9. Moreover $JMIN$ and $JMAX$ will denote the minimum and maximum values for J (i.e., here resp. 0 and 1). Finally, $Dev_{\min}$ and $Dev_{\max}$ represent the lowest and highest values for the variable Dev, i.e., $Dev_{\min} = 0$ and $Dev_{\max} = 4.5$. In fact, given a set of n statistical units, where $X_{\max}$ and $X_{\min}$ are, respectively, the maximum and minimum values among the them, then the maximum value that the standard deviation S can take is equal to (see for example Wonacott Thomas and Wonnacott Ronald 1977):

$$S^2 = \frac{(X_{\max} - X_{\min})^2}{4} \tag{6}$$

4. All peers who had graded the student who became s^+ change their SM. The model updating algorithm follows recursively a graph path starting from the s^+ students and so on backwards. So, for each learner, first K and subsequently J are updated, changing their position in the (K, J) space. Once all the students influenced by the teacher's vote have been updated, their Dev values are updated as well.

The graded learner model is updated. First the K value, and after the other values. Notice, in Eq. (5):

1. A linear function has been adopted for the variable J update, providing the two cases according to the possible value of α. In particular J_{old} could stand for J^+_{old} or J^-_{old}, depending on the student being already in S^+ (case J^+_{old}), or being just entering in S^+ (case J^-_{old}) or remaining in S^- (case J^-_{old} again);
2. In general, we assume that the assessment skill of a student depends on her *Knowledge Level* K, so the J value is a function of K. We used this type of evolutionary form as it is the easiest to treat as a first approach and also because it is used very often in automatic learning as an update of statistical variables in a machine learning context (see for example Bishop 2006).

Subsequently, the value of Dev is modified taking into account the teacher's grade. All the students, s^-, who are influenced by the graded student are modified, according to the following rules (students s^+ are fixed because graded by the teacher):

$$\begin{aligned} K^-_{\text{new}} &= K^-_{\text{grading}} + \alpha(KMAX - K^-_{\text{graded}}) \;\; (0 \le \alpha \le 1) \\ K^-_{\text{new}} &= K^-_{\text{graded}} + \alpha K^-_{\text{graded}} \;\; (\alpha < 0) \end{aligned}$$

$$\alpha = \frac{1}{IMAX}(K^-_{\text{grading}} - K^-_{\text{graded}})\frac{Dev_{\text{graded}}}{IMAX} \tag{7}$$

where K_{new} is the new value of K of the intermediate student. The ratio $\frac{Dev_{\text{graded}}}{IMAX}$ represents a kind of inertia of the value of K to change.

Each J value is changed as follows:

$$\begin{aligned}
&J^-_{\text{new}} = J^-_{\text{grading}} + \beta(JMAX - J^-_{\text{grading}}) \quad (0 \le \beta \le 1)\\
&J^-_{\text{new}} = J^-_{\text{grading}} + \beta J^-_{\text{grading}} \quad (\beta < 0)\\
&J^-_{\text{new}} = J^-_{\text{grading}} + (K^-_{\text{grading}} - K^-_{\text{graded}})\\
&(\beta = 0 \wedge J^-_{\text{grading}} = J^-_{\text{graded}})
\end{aligned}$$

with:

$$\beta = \frac{1}{IMAX}(K^-_{\text{new}} - K^-_{\text{grading}})|J_{\text{grading}} - J_{\text{graded}}|\frac{Dev_{\text{graded}}}{IMAX} \tag{8}$$

After, in order to complete the SMs, all the Dev values are updated.

4 The network learning process

In this section, we show the algorithms and the data structures that the MOOC dynamics are based on. The algorithms are based on a modified version of the *K-NN* machine learning algorithm (Mitchell 1997). K-NN is a supervised learning algorithm, whose purpose is to predict a new instance by knowing the K-data points currently in its neighbor. We use this algorithm both for its simplicity and its reduced computational time; it is possible to simulate MOOCs with thousands of students, with respect to other machine learning algorithms, such as Bayesian networks, where the computational time is unpredictable (NP-complete).

After the teacher has graded a set of s^- students, the modified K-NN algorithm can start. The learning process changes by means of the laws:

$$\begin{aligned}
&K^-_{\text{new}} = K^-_{\text{old}} + \alpha(KMAX - K^-_{\text{old}}) \quad (0 \le \alpha \le 1)\\
&K^-_{\text{new}} = K^-_{\text{old}} + \alpha(1 - K^-_{\text{old}}) \quad (\alpha < 0)\\
&\qquad \alpha = \frac{1}{I_{\max}}\frac{\sum_{i=1}^{k}\frac{1}{d_i}(K^+_i - K^-_{\text{old}})}{\sum_{i=1}^{k}\frac{1}{d_i}}\frac{Dev_i}{IMAX}
\end{aligned} \tag{9}$$

where

1. d_i is the Euclidean distance between the s^-_{old} student under update, and the ith student in the Core Group (s^+_i);
2. The K^-_{new} value is given as a convex function, to keep K in [1, 10];
3. The acronym K-NN features a K, possibly misleading here, so we are using k for the number of nearest neighbors to be used in the learning algorithm;
4. The $\frac{Dev_i}{IMAX}$ factor has the same meaning as in Eq. 7.

$$\begin{aligned}
J^-_{\text{new}} &= J^-_{\text{old}} + \frac{(K^-_{\text{new}} - K^-_{\text{old}})}{IMAX}J^-_{\text{old}} \quad (\beta = 0 \ \wedge \ J^+_i\\
&= J^-_{\text{old}}, i = 1\ldots k)\\
J^-_{\text{new}} &= J^-_{\text{old}} + \beta(JMAX - J^-_{\text{old}}) \quad (0 \le \beta \le 1)\\
J^-_{\text{new}} &= J^-_{\text{old}} + \beta J^-_{\text{old}} \quad (\beta < 0)
\end{aligned} \tag{10}$$

$$\text{with } \beta = \frac{(K^-_{\text{new}} - K_{\text{old}})}{IMAX}\frac{\sum_{i=1}^{k}\frac{1}{d_i}|J^+_i - J^-_{\text{old}}|}{\sum_{i=1}^{k}\frac{1}{d_i}}\frac{Dev_i}{IMAX}.$$

where

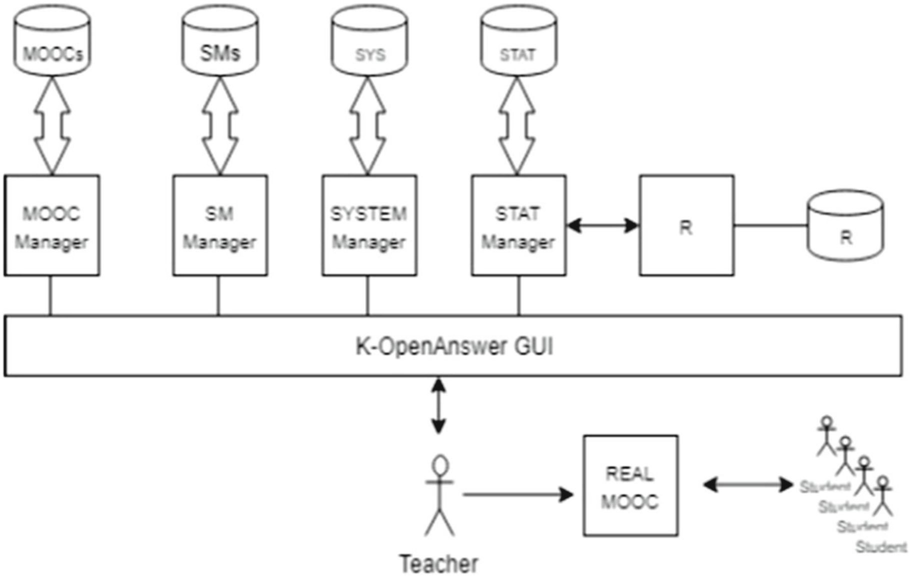
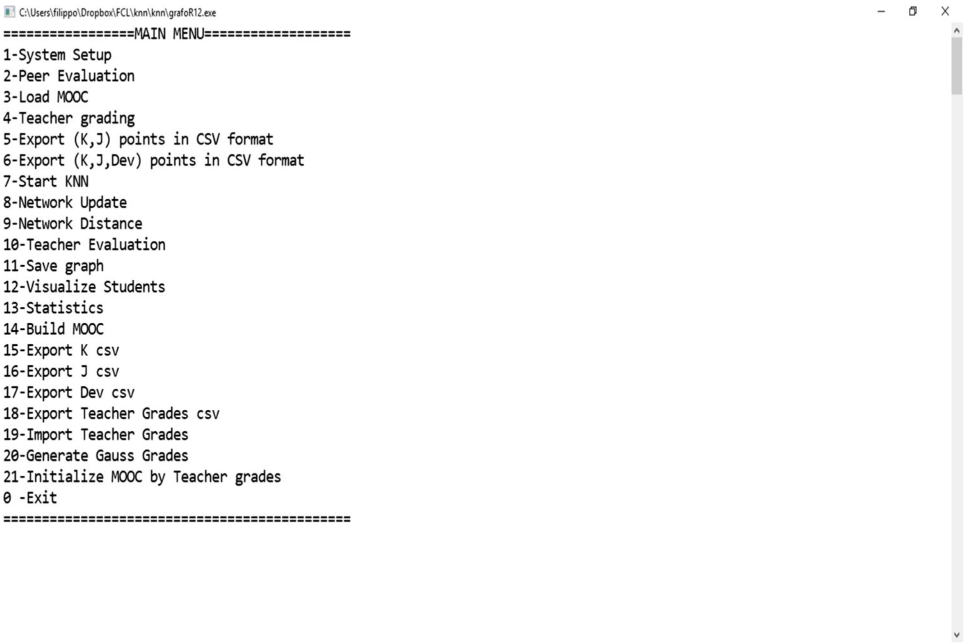


Fig. 4 Architecture of the K-OpenAnswer system



C:\Users\filippo\Dropbox\FCL\knn\knn\grafoR12.exe

```
================MAIN MENU==================
1-System Setup
2-Peer Evaluation
3-Load MOOC
4-Teacher grading
5-Export (K,J) points in CSV format
6-Export (K,J,Dev) points in CSV format
7-Start KNN
8-Network Update
9-Network Distance
10-Teacher Evaluation
11-Save graph
12-Visualize Students
13-Statistics
14-Build MOOC
15-Export K csv
16-Export J csv
17-Export Dev csv
18-Export Teacher Grades csv
19-Import Teacher Grades
20-Generate Gauss Grades
21-Initialize MOOC by Teacher grades
0 -Exit
=============================================
```

Fig. 5 Current K-OpenAnswer GUI

1. As mentioned earlier, we assume J depending on K; this is expressed through the difference between the K^-_{new} value, obtained by Eq. 9, and the K^-_{old} value;
2. d_i is the Euclidean distance between the s^-_{old} student under update, and the ith student in the Core Group (s^+_i);
3. The J_{new} value is given as a convex function, to keep J in its normal range [0, 1];
4. k is as explained in the previous equation;
5. About the coefficient β, some notices are due, for the cases when $\beta = 0$. On the one hand, when the J^+ of the k nearest neighbors is equal to the J^-_{old} value of the s^-_i student under update, J^-_{new} is computed by the difference between K^-_{new} and K^-_{old} only. The rationale is that when the s^- student changes her K^- value, her assessment skill should change as well (by the assumption of dependence of J on K). On the other hand, when the K^- value for the student under update is not changed, the assessment skill stays unchanged as well.

As can be seen from the updating rules 9 and 10, for each involved student s, the update is done by taking into consideration the K core students closest to her, calculating their distances as the Euclidean distance, (as in classic K-NN algorithm). After that, starting from this setting, the user model is updated. In the first step, the network dynamics follows the classic K-NN rule, to find which students are in the nearest neighborhood, while in the second step, the updating rules are different from those used in classic K-NN algorithm to classify new instances.

5 The K-OpenAnswer architecture

The K-OpenAnswer system in a standalone software platform is implemented in C language, to have the best computational efficiency. The overall functional architecture is shown in Fig. 4.

The system is divided into different modules, specialized to perform different functions:

- The Graphical User Interface (GUI)
- The MOOC Manager
- The STAT Manager
- The SYS Manager
- The SM Manager

5.1 The GUI

In this first version of the system, the GUI is very elementar; it appears as a list of features that the user can launch through a choice to be made with the selection of the desired number. Currently, the user can select an activity among the 21 shown in the general menu. Figure 5 shows the current GUI. Until now, there has not been much concern about the design of the GUI, having given priority to the development of the other functional modules. Obviously, the GUI represents an important aspect to have an adequate human–computer interaction. For this reason, the development of a more interactive communication environment is planned.

Table 1 MOOC Manager

Module	Goals
Build a new MOOC	A new MOOC is created
Load MOOC	A MOOC can be uploaded from a file
Export MOOC	All the students are exported into a file
Teacher Grading	The teacher can grade a group of students

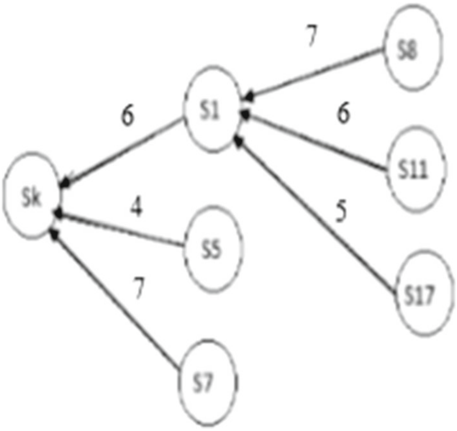


Fig. 6 An example of the MOOC graph where an arc from node S_i to node S_j represents a grade given by the student S_i to the student S_j and whose weight is the grade

5.2 The MOOC manager

This component is used to manage the simulated MOOC. The functionalities of this module are given in Table 1.

5.2.1 Build a new MOOC

Here, a new MOOC can be build. After having selected this choice, the user is requested to input the number of students and the number of peer assessments for student. A new data structure is created in a dynamic way. In our trials, we tried several configurations to test the feasibility at machine time required. The MOOC is represented by a direct and weighed graph where each node is a student and each arc from a node S_i to a node S_j represents the grade assigned from the student S_i to the student S_j, as shown in Fig. 6.

The user can leave the system possibility to completely simulate a peer evaluation; in this case, a Gaussian grades distribution among peers is used, as shown in Fig. 7, or she can manually enter the grades directly, either in online mode or in offline mode or by filling in a MOOC on file. As a result, we obtain a MOOC where each student has received n peer grades.

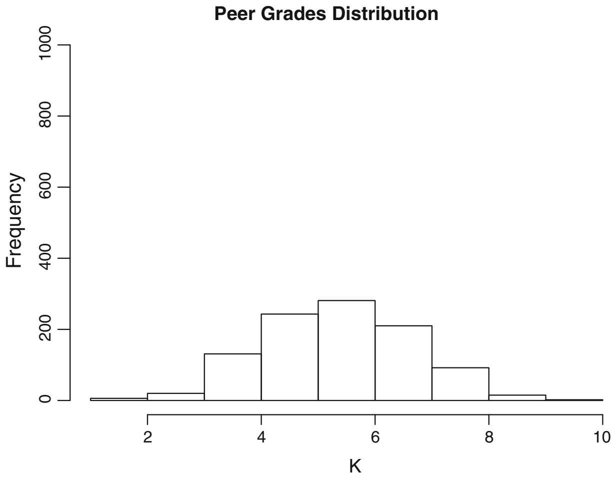


Fig. 7 An example of peer assessment distribution for a MOOC composed by 1000 students and with six peer evaluations. This is a Gaussian distribution with $\mu = 5.75$ and $\sigma = 1.74$

5.2.2 Load MOOC

This function loads a MOOC from a file. The file format is a text format representing the adjacency matrix of the MOOC. When loading the MOOC, for each student, the SM is created by means of the rules shown in Sect. 3.

5.2.3 Export MOOC

This module allows to export some MOOC variables to CSV files. In particular, one can export all the SMs, i.e., K, J and DEV variables, in group or separated. Thanks to this function, we can monitor the MOOC dynamic by other statistical programs like the R program allowing for a more in-depth study as well.

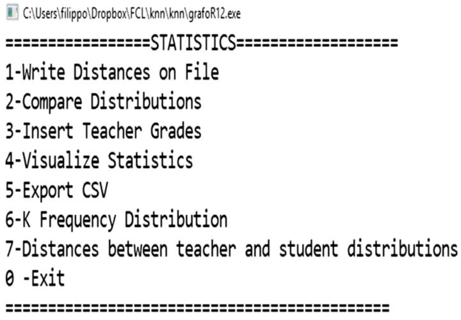
5.2.4 Teacher grading

By this option, the user (the tutor or the teacher) can grade one, all or a group of students. When a MOOC is loaded, all the students are presented to the user sorted according to their Dev value. First, this allows the user to select the students to grade, basing this choice on the reliability of their current K value as graded by peers. Secondly, the teacher can give a grade from 1 to 10.

5.3 The STAT module

The STAT module performs some statistical tasks, shown in Fig. 8.

In the following, we explain the two most significant modules.



```
C:\Users\filippo\Dropbox\FCL\knn\knn\grafoR12.exe
=================STATISTICS==================
1-Write Distances on File
2-Compare Distributions
3-Insert Teacher Grades
4-Visualize Statistics
5-Export CSV
6-K Frequency Distribution
7-Distances between teacher and student distributions
0 -Exit
============================================
```

Fig. 8 Statistics of the system

5.3.1 Distances between teacher and student distributions

This module computes the metric used for measuring the average among all students. It is the average Euclidean distance.

5.3.2 Compare distributions

This module computes the difference between the student grade distribution and the teacher grade distribution. In fact in our approach, the teacher can grade a group of students, speeding up in this way the convergence of the students' K distribution toward the one proposed by the teacher.

5.4 The SM management module

During the MOOC life cycle, the student model changes according to the K-NN algorithm. The models are stored in files and this module is responsible for managing changes. As shown in Sect. 3, each model consists of three sizes: K, J and Dev. The module is responsible for managing the persistence on file models. In particular, by this module, the user can save all the SMs in CSv or other formats, ready to be managed by specialized external programs like the R statistical package.

5.5 The overall workflow

The functionalities offered by the system are basically implementing the following workflow

- a MOOC simulation can be defined by specifying the number of students in the class and the number of peer assessments that each peer is expected to provide;
- the students in a MOOC can be given and initial distribution of models;
- a number of teacher's grades can be added into the system, determining a phase of network update (student models and grades); when n students are graded in a turn, they become members of S^+ (referring to the previous section) and the value of the Dev model variable is recalculated.
- a MOOC, represented as a direct and weighed graph, can be saved in a file for future use, and loaded back when needed; When a MOOC is loaded, students are presented to the tutor sorted according to the value of the Dev parameter of their model. This allows the teacher to possibly select answers to grade, basing the choice on students' characteristics.
- the distribution of the student models, as points on the (K,J) plan, and grades can be visualized;
- a distribution of values (Gaussian) can be used to assign grades to the students in the class.

6 Evaluation

The goals of the K-OpenAnswer system are: (i) to provide the teacher with an environment capable to simulate a MOOC ($RQ1$); (ii) to give the teacher some didactic insights about the class, coming from the evolution of the student models during a PA session. Such insights could then be applied to set up appropriate teaching strategies in a real MOOC ($RQ2$).

6.1 Research question *RQ*1

We first summarize the most important features of the system that regard the simulation of a MOOC:

- *Simulation of a big class* The teacher has the possibility to create different instances of a class, also possibly composed by thousands of learners, which is an important issue when MOOCs are the subject of study.
- *Peer Assessment* As the system allows for PA didactic strategy, where the teacher can grade a group of students in such a way to speed up the convergence of the community toward her/his grades;
- *Student modeling* Each student is represented by a model. In the current version of the system, the models are defined automatically based on a Gaussian distribution given by the teacher (through definition of mean μ and variance σ). So the teacher can select different distributions to use. In a real MOOC, the distribution might come from previous records of the class, or it can be the best fitting after trying and testing different distributions. One of the functionality of our system allows to set up the distribution of grades on the students in the class, as illustrated, by an example, in Fig. 12.
- *The statistical hypotheses* The above-mentioned Gaussian distribution can be used to simulate both PA and teacher grades distributions.
  In particular, the simulated teacher's grades are assigned based on a *Grading on a Curve* modality, i.e., a quite

Table 2 MOOC simulation trials

# Students	# Peer evaluations
100	3
500	4
1000	5
1500	3
2000	3
7000	5
10,000	5

common practice in higher education (Kulick and Wright 2008). It seeks to obtain, at the end of the grading operation, a pre-specified distribution of grades. Notice that, here, the term *curve* refers to the bell curve, the graphical representation of the probability density of the normal distribution, but this method can be used to achieve any desired distribution of the grades—for example, a uniform distribution.

- *The export functions* The system allows to extract from a PA session the data produced by the learning community during the described process (peers' grades and teacher's grades), for further processing. In particular, these data can be processed by the *R* statistical software,[5] by means of a set of dedicated functions that we built, so to allow an accurate analysis of the data produced by the MOOC; Table 1 shows a list of the above-mentioned functionalities. (Each one is implemented in a module of the software system.)
- *The learning process* The learning process is based on a modified version of the K-NN machine learning algorithm where each student is represented in the (K, J) space. Each point moves in the space until the shifts are less than a predetermined amount ϵ.

From the above features descriptions, it is possible to answer affirmatively to the first RQ; in fact, the initial state of the SMs is represented in the SMs' space; then, each grade added by the teacher (simulated grade in the simulation, yet possibly real grade in a real-world application) will make the MOOC evolve, and the new graphical representation will show "at a glance" how the SMs/points are changing positions. This is the basic feature allowing the teacher to look at the evolution of the MOOC models.

Table 2 presents a list of sample definitions of MOOC simulations used in our experimental work, while Fig. 9 illustrates the students distribution in the (K, J) space, produced by the *Visualization* menu functionality.

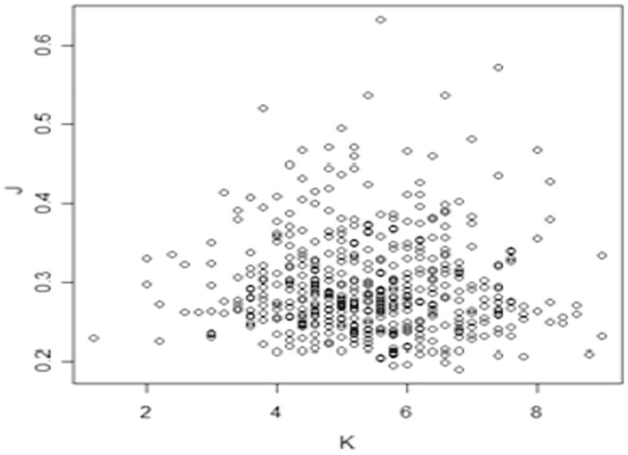


Fig. 9 MOOC with 1000 students and 3 peers evaluations

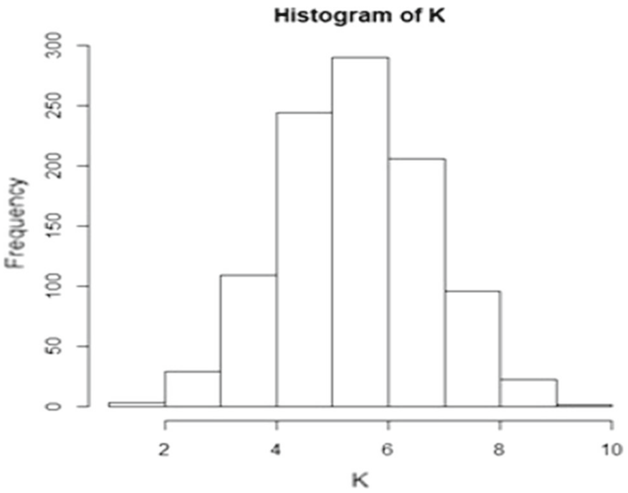


Fig. 10 An example of PA Gaussian distribution

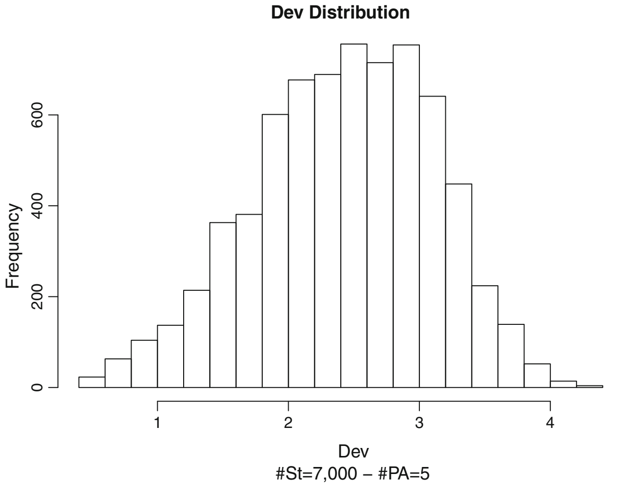


Fig. 11 Dev distribution for 10,000 students and five PA

[5] https://www.r-project.org/.

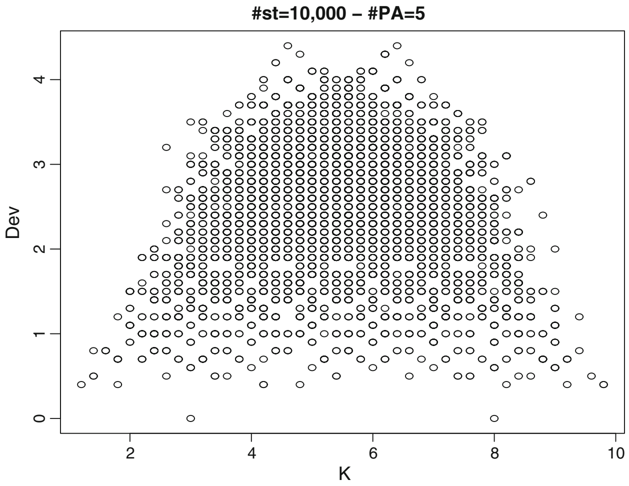


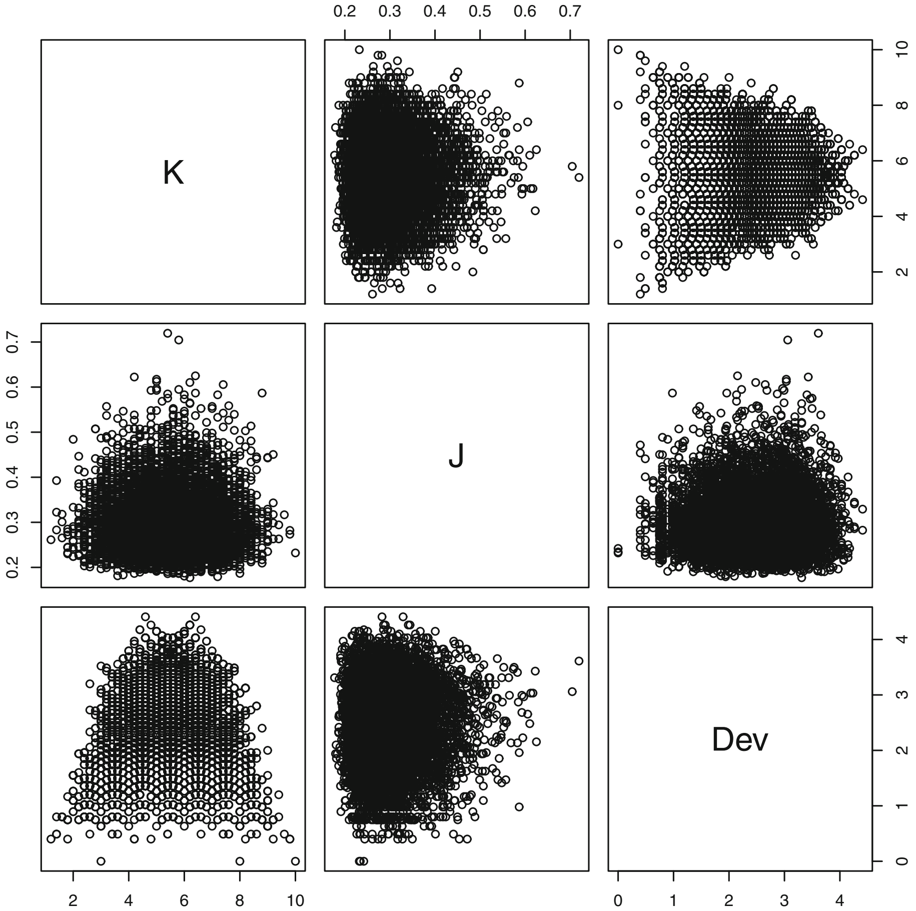
Fig. 12 (K, Dev) distribution for 10,000 students and five PA

6.2 The research question *RQ*2

As it can be expected, the answer to the second research question is supported by less evidence coming from our simulated experimentation (Figs. 10, 11). On the one hand, it is quite clear that three main insights can come from the use of the simulation system, also in the case of a real MOOC class:

1. During the teacher mediation, that is the grading activity, it is possible to select the next answer to grade basing on the visual state of the space. In particular, it is conceivable that grading the most isolated points (students) would help the system to possibly include them in the successive classification. The teacher might decide to use different criteria, still based on the visualization of the space of models allowed by our system, but on the other hand, the actual effect of such choices, and their being the best possible, can be determined only after an in-depth analysis of real-life experimentation. We think, however, that our system provides the teacher with an attractive and easy to use means, on which (s)he could capitalize, to be guided during the grading activity, even if it were sub-optimal. Figures 12 and 13 give good indications about the grading strategies to apply.
2. A second insight, coming from the use of the system, is in the possibility to appreciate how the student models evolve, as a whole, during the teacher's grading phase, and how they converge (or not) toward the

Fig. 13 A compact vision of all the SMs' variables



ideal (teacher's) grades distribution. In this sense, a non-convergence might mean that the topic of the question was too hard at this time for the class, and that some didactic preparations (further learning activities) should be undertaken before taking that topic again. Also in this case, the fact that our system can currently provide only simulations is a limit, and we can take the above consideration more as a working hypothesis that as a conclusion.

3. Finally, it seems important to underline a possible great outcome our system can offer to a teacher during a course. Several PA sessions can be held during the semester, and the evolution of the students models can be observed also considering how the model of each student evolves in time. This is a very useful source of information, made possible by the characteristics of our system, and of course, it is also quite difficult to demonstrate by means of simulations as well.

In conclusion, we can give a tentatively positive answer to the RQ2, noticing that while the system positively offers some didactic insights, the value for the teacher of such insights can be confirmed only by a real-life experimentation.

7 Conclusions and future work

In this paper, we have presented the *K-OpenAnswer* simulation environment, deemed to allow the teacher simulate and monitor the evolution of the student models in a MOOC class, during sessions of PA. The PA approach implemented by *K-OpenAnswer* is based on the K-NN method of machine learning, and provides a student modeling method representing the peers by their skill in answering to open-ended questions, and their capability to assess other peers' answers. The quality of the automated grading provided by the system for the answers is increased by using the grades the teacher assigned on a subset of the answers.

In *K-OpenAnswer*, the teacher can define a MOOC class, by giving a statistical distribution of models, and follow the simulation through the phases of (simulated) teacher's grading and student models update. During the process, the teacher can monitor the evolution of the peers models, appreciate the dynamics of the class, and get hints about the best answers to grade further, and on the reaction of the class to the given question/exercise.

The system allows also to apply the whole process in real (not simulated) cases, where the PA data, student models and grades are related to a real class and a real teacher.

We propose that this approach can allow for a reasonable applicability in MOOC contexts, and make it possible to widely adopt PA in MOOC classes.

There are limitations in the work we have presented. For one, several further analysis tools are to be implemented, such as the possibility of using different distribution policies for the initialization of the MOOCs PA data, or the support of an analysis of how many grades would be needed by the teacher to reach a safe configuration of the models, and better grade inference. These developments, as well as a pilot application to a real case of MOOC, are planned for future work.

In conclusion, we like to come back on the debate about the application of smart technologies in the city environment. It has been observed that a "normative bias" does occur in research on smart services for social ecosystems (Lytras and Visvizi 2018; Visvizi and Lytras 2018). This is the case when smart services are designed basing only on the information and communication technologies (ICT)-clouded perspective of the designer, which is often an ICT professional. This might bring a separation, and lack of correspondence, among: (i) what is implemented (decided basing on what is technically possible); (ii) what the end user might wish to have (if already conscious of such wish); (iii) the actual willingness of use, by the end user, rather than the perplexities and uneasiness about the safety, usefulness, accessibility, effectiveness and efficiency of those services. Among the advocated solutions for the above-mentioned normative bias, we think that participation is a factor, whether it is supported in a wide social network or in a more polarized social learning environment. In this paper, we have declined participation in the context of education, implemented as the PA activity performed by students in a large class. PA has several different declinations, from the sole answer to a question, and grading of some peers' answers, through activities of production and assessment of projects, with or without interaction by means of social exchange, to the performance and assessment of group work. PA is a great formative tool too, and can be extended to application involving ecosystem-wide training on demand. An interesting item for future work is then in the study of the extension of our framework to use in bigger social groups, for educational and exchange purposes. In this activity, we plan to import the research efforts done in the past with respect to the community of practice (CoP) concept (Wenger 2010), whereas a CoP presents itself with characteristics of social networks, and of exchange environments where learning can be ignited.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Human and animal rights This article does not contain any studies with human participants performed by any of the authors. This article does not contain any studies with animals performed by any of the authors.

References

Anderson LW, Krathwohl DR (2000) A taxonomy for learning, teaching, and assessing: a revision of Bloom's taxonomy of educational objectives. Allyn and Bacon, Boston

Anson R, Goodman JA (2014) A peer assessment system to improve student team experiences. J Educ Bus 89(1):27–34

Bishop CM (2006) Pattern recognition and machine learning (information science and statistics). Springer, Berlin

Bloom B, Engelhart M, Furst E, Hill W, Krathwohl D (1956) Taxonomy of educational objectives: the classification of educational goals. Handbook I: cognitive domain. McGraw-Hill, New York

Class-central mooc report. https://www.classcentral.com/report/mooc-stats-2018/

De Marsico M, Sciarrone F, Sterbini A, Temperini M (2017a) Supporting mediated peer-evaluation to grade answers to open-ended questions. EURASIA J Math Sci Technol Educ 13(4):1085–1106

De Marsico M, Sterbini A, Sciarrone F, Temperini M (2017b) Modeling a peer assessment framework by means of a lazy learning approach. In: Huang TC, Lau R, Huang YM, Spaniol M, Yuen CH (eds) Emerging technologies for education. Springer, Cham, pp 336–345

De Marsico M, Sciarrone F, Sterbini A, Temperini M (2018) Peer assessment and knowledge discovering in a community of learners. In: IC3K 2018—Proceedings of the 10th international joint conference on knowledge discovery, knowledge engineering and knowledge management, pp 119–126

Kane LS, Lawler EE (1978) Methods of peer assessment. Psychol Bull 85:555–586

Kizilcec RF, Schneider E, Cohen GL, McFarland DA (2014) Encouraging forum participation in online courses with collectivist, individualist and neutral motivational framings. eLearning Pap 37:13–21

Kulick G, Wright R (2008) The impact of grading on the curve: a simulation analysis. Int J Scholarsh Teach Learn 2(5):2

Li L, Liu X, Steckelberg AL (2010) Assessor or assessee: how student learning improves by giving and receiving peer feedback. Br J Educ Technol 41:525–536

Limongelli C, Sciarrone F, Vaste G (2008) Ls-plan: an effective combination of dynamic courseware generation and learning styles in web-based education. In: Proceedings of the AH'08: adaptive hypermedia and adaptive web-based systems, vol. 5149 LNCS, pp 133–142

Limongelli C, Lombardi M, Marani A, Sciarrone F (2013) A teacher model to speed up the process of building courses. In: Proceedings of the 15th international conference on human-computer interaction. Applications and services, HCI International 2013, Las Vegas, NV, USA, July 21–26, 2013, Part II, pp 434–443

Limongelli C, Sciarrone F, Temperini M (2015) A social network-based teacher model to support course construction. Comput Hum Behav 51:1077–1085

Lytras M, Visvizi A (2018) Who uses smart city services and what to make of it: toward interdisciplinary smart cities research. Sustainability 10:1998–2013

Lytras M, Visvizi A, Daniela L, Sarirete A, Ordonez De Pablos P (2018) Social networks research for sustainable smart education. Sustainability 10:2974–2993

Metcalfe J, Shimamura A (1994) Metacognition: knowing about knowing. MIT Press, Cambridge

Mitchell TM (1997) Machine learning, 1st edn. David McKay, New York

Onah D, Sinclair J, Boyatt R, Foss J (2014) Massive open online courses: learner participation. In: Proceeding of the 7th international conference of education, research and innovation, pp 2348–2356

Palmer K, Richardson P (2003) On-line assessment and free-response input—a pedagogic and technical model for squaring the circle. In: Proceedings of the 7th CAA conference, pp 289–300

Penstein Rosé C, Ferschke O (2016) Technology support for discussion based learning: from computer supported collaborative learning to the future of massive open online courses. Int J Artif Intell Educ 26(2):660–678

Piech C, Huang J, Chen Z, Do CB, Ng AY, Koller D (2013) Tuned models of peer assessment in moocs. In: EDM

Reynolds J, Moskovitz C (2008) Calibrated peer review assignments in science courses: are they designed to promote critical thinking and writing skills? J Coll Sci Teach 38(2):60–66

Sadler P, Good E (2006) The impact of self- and peer-grading on student learning. Educ Assess 11(1):1–31

Sterbini A, Temperini M (2012a) Dealing with open-answer questions in a peer-assessment environment. In: Proceedings of the ICWL 2012. LNCS, vol 7558, pp 240–248

Sterbini A, Temperini M (2012b) Supporting assessment of open answers in a didactic setting. In: 2012 IEEE 12th international conference on advanced learning technologies (ICALT), pp 678–679

Sterbini A, Temperini M (2013) Analysis of open answers via mediated peer-assessment. In: 2013 17th international conference on system theory, control and computing (ICSTCC), pp 663–668

Suen HK (2014) Peer assessment for massive open online courses (moocs). Int Rev Res Open Distance Learn 15(3):312–327

Tenório T, Bittencourt II, Isotani S, Silva AP (2016) Does peer assessment in on-line learning environments work? A systematic review of the literature. Comput Hum Behav 64:94–107

Visvizi A, Lytras M (2018) Rescaling and refocusing smart cities research: from mega cities to smart villages. J Sci Technol Policy Manag 9(2):134–145

Wenger E (2010) Communities of practice and social learning systems: the career of a concept. Springer, London, pp 179–198

Wonacott Thomas H, Wonnacott Ronald J (1977) Introductory statistics. Wiley, New York

Yang D, Adamson D, Rosé CP (2014) Question recommendation with constraints for massive open online courses. In: Proceedings of the 8th ACM conference on recommender systems, RecSys'14. ACM, New York, NY, USA, pp. 49–56. https://doi.org/10.1145/2645710.2645748

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.