Proceedings of the Twenty-Sixth International Florida Artificial Intelligence Research Society Conference

Applying Clustering to the Problem of Predicting Retention within an ITS: Comparing Regularity Clustering with Traditional Methods

Fei Song, Shubhendu Trivedi, Yutao Wang, Gábor N. Sárközy and **Neil T. Heffernan**
Worcester Polytechnic Institute
TTI at Chicago

Abstract

In student modeling, the concept of "mastery learning" i.e. that a student continues to learn a skill till mastery is attained is important. Usually, mastery is defined in terms of most recent student performance. This is also the case with models such as Knowledge Tracing which estimate knowledge solely based on patterns of questions a student gets correct and the task usually is to predict immediate next action of the student. In retrospect however, it is not clear if this is a good definition of mastery since it is perhaps more useful to focus more on student retention over a longer period of time. This paper improves a recently introduced model by Wang and Beck that predicts long term student performance by clustering the students and generating multiple predictions by using a recently developed ensemble technique. Another contribution is that we introduce a novel clustering algorithm we call "Regularity Clustering" and show that it is superior in the task of predicting student retention over more popular techniques such as *k*-means and Spectral Clustering.

Introduction

An important concept in student modelling is of "mastery learning" - that is, a student continues to learn a skill till mastery is achieved. While the exact definition of mastery varies, it is usually defined in terms of the most recent student performance. For example, in the Knowledge Tracing (Corbett and Anderson, 1995) framework that has come to dominate student modelling in many contexts, mastery in a skill is said to have been achieved when according to the model the probability that the student knows the skill exceeds 0.95. In many actual tutoring systems this definition is relaxed but still relies on the idea of recent performance. In a recent work (Wang and Beck, 2012) draw our attention to the question whether such a near singular focus is important after all. Intuitively, whether a student will remember enough to answer a question after taking a break is a better definition of mastery as compared to a local measure based on next item response. That is, they found that features such as the number of distinct days that the student practised a skill was more important than features that accounted for how many questions they got correct. It is noteworthy that models such as Knowledge Tracing are in stark contrast to this, they only rely on the patterns of questions that students get correct or incorrect to make a prediction of their response on the next item, and hence factors such as how many questions they get correct are more important. This difference is not surprising since the factors that reflect long term retention might be quite different from factors that cause good short term performance.

In retrospect, it is probably unfortunate that the Intelligent Tutoring Systems field has fallen into using the term "mastery" when that often meant "demonstrated some retention over a few minutes". Koedinger argues that we need "robust" learning from our tutors, and being able to demonstrate retention days later is clearly a more robust notion of learning that immediate retention after practice.

To attempt to improve upon Wang & Beck, we have used the technique of using clustering to generate an ensemble introduced by (Trivedi, Pardos and Heffernan, 2011) to see if we can improve our predictions. The first research question that we have is: Can we employ this technique to increase accuracy in predicting long term retention? In (Trivedi, Pardos, Sárközy and Heffernan, 2011) it was found that Spectral clustering was more effective than K-means for this type of work. We also introduce and test a novel type of clustering that we call "Regularity Clustering", which is derived from the Regularity Lemma (Szemerédi, 1976), a fundamental result in graph theory. We also ask the following empirical question: "How does Regularity Clustering compare in performance with spectral and K-means?" In the next section we review a technique that uses clustering for bootstrapping.

Clustering Students and Strategy for Bootstrapping

The idea that students are perhaps quite different when it comes to forgetting makes it quite apparent that it is perhaps not a good idea to fit a global model on all of the data. In spite of individual differences, it is well known to teachers that broadly the patterns and underlying reasons of forgetting fall into several coarse groups, with each such group having students more "similar" to each other in regard to forgetting. Honing on this intuition, it might make more sense to cluster students into somewhat homogeneous groups and then train a predictor separately on each such group, which considers only the points from that cluster as the training set

for itself. It is clear that each such predictor would be a better representative for that group of students as compared to a single global predictor trained on all the students at one time. While this idea sounds compelling, there is a major issue with it. While it is useful to model students as belonging to different groups, it is perhaps not a good idea to simply divide them into clusters. This is because the groupings are usually not very clear. For example, a student might be extremely good at retaining information about certain aspects of Trigonometry but not other aspects, while at the same time might be strong with retaining algebra. Such complex characteristics can not be modelled by a simplistic solution as only clustering the data to some upper limit and then training predictors on each cluster. The "fuzzy" nature of such a process, which is like a spread of features across groups needs to be captured to make a distributive model such as the above more meaningful. This issue can be fixed by varying the granularity of the clustering and training separate models each time so the such features can be accounted for. A simple strategy to do so was proposed recently and was found quite useful in various tasks in student modelling (Trivedi, Pardos and Heffernan, 2011), (Trivedi, Pardos, Sárközy and Heffernan, 2011).

The technique is actually a simple ensemble method. The basic idea behind ensemble methods is that they involve running a "base learning algorithm" multiple times, each time with some change in the representation of the input (e.g. considering only a subset of the training examples or a subset of features etc) so that a number of diverse predictions can be obtained. This process also gives a rich representation of the input, which is one of the reasons why they work so well. In the particular case of our method, unlike many other ensemble methods that use a random subset to bootstrap, we use clustering to bootstrap. The training set is first clustered into k disjoint clusters and then a logistic regression model is trained on each of the clusters only based on the training points that were assigned to that cluster. Each such model, being a representative of a cluster is referred to as a *cluster model*. Thus for a given value of k there would be k cluster models. Note that since all the clusters are mutually exclusive, the training set is represented by all the k *cluster models* taken together. We refer to this as a *Prediction Model*, PM_k. For an incoming test point, we first figure out the cluster that point belongs to and then use the concerned cluster model alone to make a prediction on that point. Now also note that we don't specify the number of clusters above. Hence, we can change the granularity of the clustering from 1 (PM_1, which is the entire dataset as one cluster) to some high value K. In each such instance we would get a different *Prediction Model*, thus obtaining a set of K *Prediction Models*. Since the granularity of the clustering is varied, the predictions obtained would be diverse and hence could be combined together by some method such as averaging them together to get a single prediction.

Note that the clustering algorithm above is not specified and hence could be any clustering technique, as long as there is a straightforward way to map test points to clusters. In particular we clustered students using three algorithms: *k*-means (Hartigan *et al* 1979), Spectral Clustering (Luxburg, 2007) and a recently introduced clustering technique called Regularity Clustering (Sárközy, Song, Szemerédi and Trivedi, 2012). The basic *k*-means algorithm finds groupings in the data by randomly initializing a set of k cluster centroids and then iteratively minimizing a distortion function and updating these k cluster centroids and the points assigned to them. This is done till a point is reached such that sum of the distances of all the points with their assigned cluster centroids is as low as possible. Clustering methods such as *k*-means estimate explicit models of the data (specifically spherical Gaussians) and fail spectacularly when the data is organized in very irregular and complex shaped clusters. Spectral clustering on the other hand works quite differently. It represents the data as an undirected graph and analyses the spectrum of the graph Laplacian obtained from the pairwise similarities of the data points (also called the similar matrix of the graph). This view is useful as it does not estimate any explicit model of the data and instead works by unfolding the data manifold to form meaningful clusters. Usually spectral clustering is a far more accurate clustering method as compared to k-means except in cases where the data indeed confirms to the model that the k-means estimates. For more details we refer the reader the mentioned references. In the next section we describe the newly introduced clustering technique called Regularity Clustering. This section might be skipped to experimental results without any loss of generality.

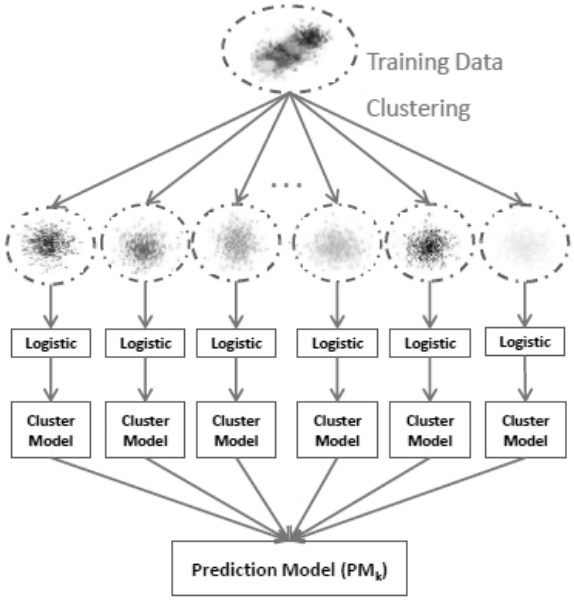

Figure 1: Construction of a Prediction Model for a given K. See text for details

Regularity Clustering Algorithm

In this section we briefly describe the Regularity Clustering algorithm. The Regularity Lemma (Szemerédi, 1976) is a fundamental result in Graph Theory that claims the existence of a regular partition of the vertex set of the graph

(the actual definition of "regular" follows), from which we can construct a reduced graph, this reduced graph is an "essence" of the original graph and can be worked on instead of the original graph. The Regularity Lemma is a very important tool in theoretical proofs, but due to the requirement of very large graph, it doesn't have practical applications. A clustering method was recently introduced that makes an attempt to harness the power of the Regularity Lemma(Sárközy, Song, Szemerédi and Trivedi, 2012). Before we describe the algorithm, we first introduce some notation.

Notation and Definitions

Let $G = (V, E)$ denote a graph, where V is the set of vertices and E is the set of edges. When A, B are disjoint subsets of V, the number of edges with one endpoint in A and the other in B is denoted by $e(A, B)$. When A and B are nonempty, we define the *density* of edges between A and B as

$$d(A, B) = \frac{e(A, B)}{|A||B|}.$$

The most important concept is the following.

Definition 1 *The bipartite graph $G = (A, B, E)$ is ϵ-regular if for every $X \subset A$, $Y \subset B$ satisfying*

$$|X| > \epsilon|A|, |Y| > \epsilon|B|$$

we have

$$|d(X, Y) - d(A, B)| < \epsilon,$$

otherwise it is ϵ-irregular.

Roughly speaking this means that in an ϵ-regular bipartite graph the edge density between *any* two relatively large subsets is about the same as the original edge density. In effect this implies that all the edges are distributed almost uniformly.

Definition 2 *A partition P of the vertex set $V = V_0 \cup V_1 \cup \ldots \cup V_k$ of a graph $G = (V, E)$ is called an equitable partition if all the classes $V_i, 1 \leq i \leq k$, have the same cardinality. V_0 is called the exceptional class.*

Thus note that the exceptional class V_0 is there only for a technical reason, namely to guarantee that the other classes have the same cardinality.

Definition 3 *An equitable partition P of the vertex set $V = V_0 \cup V_1 \cup \ldots \cup V_k$ of $G = (V, E)$ is called ϵ-regular if $|V_0| < \epsilon|V|$ and all but ϵk^2 of the pairs (V_i, V_j) are ϵ-regular where $1 \leq i < j \leq k$.*

The Regularity Lemma basically claims that every (dense) graph could be partitioned into a bounded number of pseudo-random bipartite graphs and a few leftover edges. Since random graphs of a given edge density are much easier to treat than all graphs of the same edge-density, the Regularity Lemma helps us to translate results that are trivial for random graphs to the class of all graphs with a given number of edges.

In applications of the Regularity Lemma the concept of the *reduced graph* plays an important role.

Definition 4 *Given an ϵ-regular partition of a graph $G = (V, E)$ as provided by the Regularity Lemma, we define the reduced graph G^R as follows. The vertices of G^R are associated to the classes in the partition and the edges are associated to the ϵ-regular pairs between classes with density above d.*

The reduced graph would preserve most of the properties of the original graph (see (Komlós *et al.*, 2002)). This implies that if we run any algorithm on G^R instead of G we would get a significant speed-up without compromising accuracy much.

Algorithmic Version of the Regularity Lemma

The original proof of the regularity lemma (Szemerédi, 1976) does not give a method to construct a regular partition but only shows that one must exist. To apply the regularity lemma in practical settings, we need a constructive version. Alon *et al.* (Alon *et al.*, 1994) were the first to give an algorithmic version. Since then a few other algorithmic versions have also been proposed (Frieze and Kannan, 1999). Below we give a brief description to the algorithm due to Alon *et al*, The details can be found at (Alon *et al.*, 1994).

First, given a pair (A, B), they have a subroutine (Lemma1) which can either verify that the pair is ϵ-regular or provide a certificate that it is not. The certificate is the subset $(A', B') \subset (A, B)$ and it helps to proceed to the next step in the algorithm.

So given a concrete partition, the algorithm can check the regularity of each pair by using Lemma1. If there are enough regular pairs then the algorithm terminated with the conclusion that this is indeed a regular partition. Otherwise by using all the certificates found before, the algorithm divides each class into a set of "atoms", then splitting each atom into a set of equal sized classes. By doing so on the original partition, the algorithm forms a new partition which is guranteed to have better chance to be a regular partition.

Alon *et al* proved that the algorithm must halt after certain iterations (see (Alon *et al.*, 1994)). Unfortunately, the number of iterations is huge, also in each iteration the number of classes increases to $k4^k$ from k, starting from some integer $k > 1$. This implies that the graph G must be indeed astronomically large (a tower function) to ensure the completion of this procedure. As mentioned before, Gowers (Gowers, 1997) proved that indeed this tower function is necessary in order to guarantee an ϵ-regular partition for *all* graphs. The size requirement of the algorithm above makes it impractical for real world situations where the number of vertices typically is a few thousand.

Spectral regularity algorithm

To make the regularity lemma applicable we first needed a constructive version that we stated above. But we see that even the constructive version is not directly applicable to real world scenarios. We note that the above algorithm has such restrictions because its aim is to find a perfect regular partition. Thus, to make the regularity lemma truly applicable, we modify the Regular Partition Algorithm so that instead of constructing a regular partition, we find an *approx-*

imately regular partition. Such a partition should be much easier to construct. We have the following 3 major modifications to the Regular Partition Algorithm given by Alon *et al*.

Modification 1: We want to decrease the cardinality of atoms in each iteration. Instead of using all the ϵ-irregular pairs, we only use some of them. Specifically, in our current implementation, for each class we consider at most one ϵ-irregular pair that involves the given class. By doing this we reduce the number of atoms to at most 2. We observe that in spite of the crude approximation, this seems to work well in practice.

Modification 2: We want to bound the rate by which the class size decreases in each iteration. In the original Refinement Algorithm, each class will be divided into 4^k subclasses. Since Modification 1 gurantees at most 2 atoms for each class, we could significantly decrease the number of subclasses to l, where a typical value of l could be 3 or 4, much smaller than 4^k. We call this user defined parameter l the refinement number.

Modification 3: In the original Refinement Algorithm the exceptional class is guranteed to be small. Our Modification 2 might cause the size of the exceptional class to increase too fast. Indeed, by using a smaller l, we risk putting $\frac{1}{l}$ portion of all vertices into V_0 after each iteration. To overcome this drawback, we "recycle" most of V_0, i.e. we move back most of the vertices from V_0. Below is the modified Refinement Algorithm.

Modified Refinement Algorithm: *Given a γ-irregular equitable partition P of the vertex set $V = V_0 \cup V_1 \cup \ldots \cup V_k$ with $\gamma = \frac{\epsilon^4}{16}$ and refinement number l, construct a new partition Q.*
For each pair (V_s, V_t), $1 \leq s < t \leq k$, we apply Lemma1 with $A = V_s$, $B = V_t$ and ϵ. For a fixed s if (V_s, V_t) is found to be ϵ-regular for all $t \neq s$ we do nothing, i.e. V_s is one atom. Otherwise, we select one ϵ-irregular pair (V_s, V_t) randomly and the corresponding certificate partitions V_s into two atoms. Set $m = \lfloor \frac{|V_i|}{l} \rfloor$, $1 \leq i \leq k$. Then we choose a collection Q' of pairwise disjoint subsets of V such that every member of Q' has cardinality m and every atom A contains exactly $\lfloor \frac{|A|}{m} \rfloor$ members of Q'. Then we unite the leftover vertices in each V_s, we select one more subset of size m from these vertices and add these sets to Q' resulting in the partition Q. The collection Q is an equitable partition of V into at most $1 + lk$ classes.

Now we present our modified Regular Partition Algorithm. There are three main parameters to be selected by the user: ϵ, the refinement number l and h the minimum class size when we must halt the refinement procedure. h is used to ensure that if the class size has gone too small then the procedure should not continue.

Modified Regular Partition Algorithm :

Given a graph G and parameters ϵ, l, h, construct an approximately ϵ-regular partition.

1. **Initial partition:** *Arbitrarily divide the vertices of G into an equitable partition P_1 with classes $V_0, V_1, \ldots, V_l$, where $|V_1| = \lfloor \frac{n}{l} \rfloor$ and hence $|V_0| < l$. Denote $k_1 = l$.*
2. **Check size and regularity:** *If $|V_i| < h$, $1 \leq i \leq k$, then halt. Otherwise for every pair (V_s, V_t) of P_i, verify if it is ϵ-regular or find $X \subset V_s, Y \subset V_t, |X| \geq \frac{\epsilon^4}{16}|V_s|, |Y| \geq \frac{\epsilon^4}{16}|V_t|$, such that $|d(X,Y) - d(V_s, V_t)| \geq \epsilon^4$.*
3. **Count regular pairs:** *If there are at most ϵk_i^2 pairs that are not verified as ϵ-regular, then halt. P_i is an ϵ-regular partition.*
4. **Refinement:** *Otherwise apply the Modified Refinement Algorithm, where $P = P_i, k = k_i, \gamma = \frac{\epsilon^4}{16}$, and obtain a partition Q with $1 + lk_i$ classes.*
5. **Iteration:** *Let $k_{i+1} = lk_i, P_{i+1} = Q, i = i + 1$, and go to step 2.*

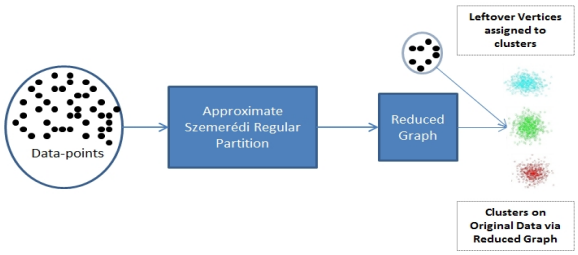

Figure 2: A Two Phase Strategy for Clustering

Two Phase Strategy

To make the regularity lemma applicable for clustering data, we still need to resolve two issues: Firstly, in practise we don't require equitable partition; and secondly, we do not have full control on the number of clusters in the final partition. To overcome these, we adopt the following two phase strategy (Figure 1):

1. **Application of the Regular Partition Algorithm:** In the first stage we apply the regular partition algorithm as described in the previous section to obtain an approximately regular partition of the graph representing the data. Once such a partition has been obtained, the reduced graph as described in Definition 4 could be constructed from the partition.
2. **Clustering the Reduced Graph:** We apply spectral clustering (though any other pairwise clustering technique could be used) on the reduced graph to get a partitioning and then project it back to the higher dimension. Recall that vertices in the exceptional set V_0, are leftovers from the refinement process and must be assigned to the clusters obtained. Thus in the end these leftover vertices are redistributed amongst the clusters using *knn* classifier to get the final grouping.

In the next section we discuss the dataset considered and the results.

Dataset Description and Experimental Results

The data considered in this article comes from the ASSISTments system, a web-based tutoring system for 4th to 10th

grade mathematics. The system is widely used in Northeastern United States by students in labs and for doing homework in the night. The dataset used is the same as used in (Wang and Beck, 2012). The only exception being that we considered the data for a unique 1969 students and did not consider multiple data points of the same student attempting something from a different skill. This was only done because we were interested in clustering students according to user-id. The following features were used. The goal was to predict whether a response was correct i.e. 1 or incorrect or 0.

1. *n_correct:* the number of prior student correct responses on this skill; This feature along with *n_incorrect*, the number of prior incorrect responses on this skill are both used in PFA models.
2. *n_day_seen:* the number of distinct days on which students practiced this skill. This feature distinguishes the students who practiced more days with fewer opportunities each day from those who practiced fewer days but more intensely, and allow us to evaluate the difference between these two situations. This feature was designed to capture certain spaced practice effect in students data.
3. *g_mean_performance:* the geometric mean of students previous performances, using a decay of 0.7. For a given student and a given skill, use opp to represent the opportunity count the student has on this skill, we compute the geometric mean of students previous performance using formula: $g\_mean\_performance(opp) = g\_mean\_performance(opp - 1) \times 0.7 + correctness(opp) \times 0.3$. The geometric mean method allows us to examine current status with a decaying memory of history data. The number 0.7 was selected based on experimenting with different values.
4. *g_mean_time:* the geometric mean of students previous response time, using a decay of 0.7. Similar with g_mean_performance, for a given student and a given skill, the formula of the geometric mean of students previous response time is: $g\_mean\_time(opp) = g\_mean\_time(opp - 1) \times 0.7 + response\_time(opp) \times 0.3$.
5. *slope_3:* the slope of students most recent three performances. The slope information helps capture the influence of recent trends of student performance.
6. *delay_since_last:* the number of days since the student last saw the skill. This feature was designed to account for a gradual forgetting of information by the student.
7. *problem_difficulty:* the difficulty of the problem. The problem_difficulty term is actually the problem easiness in our model, since it is represented using the percent correct for this problem across all students. The higher this value is, the more likely the problem can be answered correctly.

Out of these features it was reported that features such as *n_correct* and *n_incorrect* had very little influence on the prediction performance while the features *g_mean_performance* and *n_day_seen* appear to be reliable predictors of student retention. This observation is consistent with the spaced practice effect in cognitive science. Hence, in our experiments we don't consider *n_correct* and *n_incorrect* while training the model. As mentioned before, we used *k*-means, Spectral and Regularity Clustering in conjunction with the ensemble technique described. It must also be noted that the features were normalized to values between -1 and 1 to avoid undue dominance of performance by a specific feature. The results obtained were rather surprising. The use of *k*-means clustering and Spectral Clustering, that has been reported useful in other tasks does not seem to help in the case of predicting long term retention (atleast on this data). The baseline model used by Wang & Beck is represented in Figure 3 by PM_1, the starting point on the x-axis. The other values on the x-axis represent how many *Prediction Models* were averaged. The errors reported are the mean absolute errors. As reported in Table 1, the ensemble used in conjunction with Regularity Clustering is significantly better than the baseline with strong p-values. In table 2 we show that this trend also holds when Regularity Clustering is compared with Spectral Clustering.

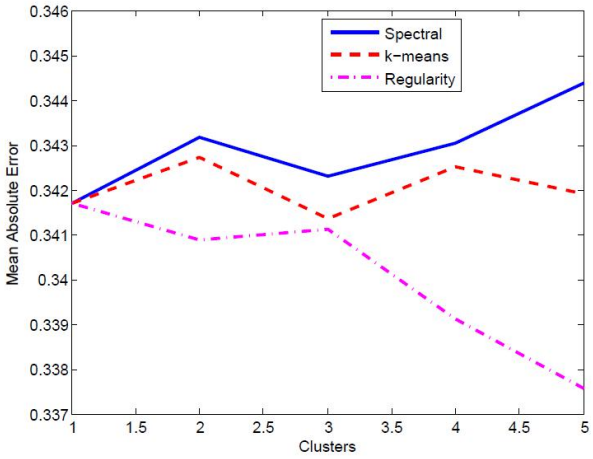

Figure 3: Mean Absolute Errors on Using the three Clustering Techniques for Bagging

Table 1: Paired t-tests on the predictions obtained with the baseline (PM_1) and Regularity Clustering

Pred. Models	Baseline & Regularity
1	-
2	0.00531
3	**0.0401**
4	**0.0018**
5	**0.0044**

Conclusion

(Corbett and Anderson, 1995) found time and again that Knowledge Tracing was consistently over predicting student performance on paper and pencil measures, and we suggest

Table 2: Paired t-tests on the predictions obtained with Spectral and with Regularity Clustering at different k

Pred. Models	Spectral & Regularity
1	-
2	0.1086
3	**0.0818**
4	**0.0045**
5	$\ll$ **0.005**

that a focus on retention days later might be a way to correct that. This paper makes two contributions, one is in educational data mining and another is a contribution to the literature on clustering. We use this new clustering technique to help predict student long term retention and compared the result of different clustering techniques. From the results, we can draw two important conclusions: Firstly, by adding the ensemble cluster model technique build in (Trivedi, Pardos and Heffernan, 2011), we are able to improve the model that Wang and Beck used to predict student long-term retention; Another conclusion is that, the Regularity Clustering method provides reliably higher predictive accuracy in predict student retention compare to other popular existing clustering methods we used: K-means and the spectral clustering. The paper shows the clear value of clustering in predicting student long-term retention lending weight to the fact that the patterns of forgetting in students might roughly be in groups. This paper also suggests that Regularity Clustering is the most effective clustering method in this task, atleast in the methods that we tried, outperforming techniques such as Spectral Clustering.

References

A. T Corbett and J. R. Anderson. 1995. Knowledge Tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, 4: pp. 253-278.

N. Alon, R. A. Duke, H. Lefmann, V. Rödl, R. Yuster. 1994. The Algorithmic Aspects of the Regularity Lemma. *Journal of Algorithms*, 16, pp. 80-109.

A. M. Frieze, R. Kannan. 1999. A simple algorithm for constructing Szemerédi's regularity partition. *Elec. J. Comb*, 6.

W. T. Gowers. 1997. Lower bounds of tower type for Szemerédi's uniformly lemma. Geom. Funct. Anal 7, pp. 322-337.

J. A. Hartigan, M. A. Wong. 1979. Algorithm AS 136: A K-Means Clustering Algorithm, In *J Royal Stat. Soc. Series C (App. Stat.)*, 28 (1): pp. 100108.

J. Komlós, A. Shokoufandeh, M. Simonovits, and E. Szemerédi. 2002. The Regularity Lemma and Its Applications in Graph Theory. *Theoretical Aspects of Computer Science*, LNCS 2292, pp. 84-112.

U. Luxburg A Tutorial on Spectral Clustering, In *Statistics and Computing*, Kluwer Academic Publishers, Hingham, MA, USA, Vol 17, Issue 4, 2007.

G. N. Sárközy, F. Song, E. Szemerédi and S. Trivedi. A Practical Regularity Partitioning Algorithm and its Applications in Clustering. arXiv preprint arXiv:1209.6540.

E. Szemerédi, Regular partitions of graphs, Colloques Internationaux C.N.R.S. N^{o} 260 - *Problèmes Combinatoires et Théorie des Graphes*, Orsay (1976), pp. 399-401.

S. Trivedi, Z. A. Pardos and N. T. Heffernan. 2011. Clustering Students to Generate an Ensemble to Improve Standard Test Predictions, The fifteenth international Conference on Artificial Intelligence in Education.

S. Trivedi, Z. A. Pardos, G. Sarkozy and N. T. Heffernan. 2011. Spectral Clustering in Educational Data Mining.Proceedings of *the 4th International Conference on Educational Data Mining*, pp. 129-138.

Y. Wang and J. E. Beck. 2012. Incorporating Factors Influencing Knowledge Retention into a Student Model. In the Proceedings of *the 5th International Conference on Educational Data Mining*, pp. 201-203.