Predictive Sequential Pattern Mining via Interpretable Convolutional Neural Networks

Lan Jiang
University of Illinois Urbana–Champaign
Champaign, IL, USA
lanj3@illinois.edu

Nigel Bosch
University of Illinois Urbana–Champaign
Champaign, IL, USA
pnb@illinois.edu

ABSTRACT

We present an algorithm using interpretable convolutional neural networks for mining sequential patterns from event log data. The key to our approach is utilizing structured regularization to achieve sparse parameter values that closely resemble the results of typical pattern mining algorithms, and allows the learned convolution filters to be interpreted easily. Our method can handle both sequences of individual, unique elements and concurrent multiple-element sequences, which represents most situations where sequences may occur in logs of student actions. We applied our structured regularization method to a self-supervised problem predicting future actions from past actions in two different educational datasets as example applications. Furthermore, we generated features from the learned patterns to evaluate the utility of patterns and trained a supervised model with these features to predict academic outcomes via transfer learning. Our algorithm improves the correlation of sequences with outcomes by an average of $r = .131$ on one dataset and $r = .101$ on the other dataset versus a traditional sequential pattern mining algorithm. Finally, we visualize the extracted patterns and demonstrate that they can be interpreted as a sequence of actions.

Keywords

Interpretability, pattern mining, convolutional neural networks, sequential data

1. INTRODUCTION

Convolutional neural networks (CNNs) have been successfully applied to various applications in educational data mining [2, 9, 15, 16, 22, 24]. However, CNNs have a major shortcoming in terms of transparency, because they typically form "end to end" models that make high-level inferences from low-level inputs through a series of opaque layers. Thus, resulting models are often hard to understand and interpret. As a consequence, both instructors and students do not know what kinds of student behaviors actually impact predictions, which is important for understanding and supporting students' learning behaviors and instructors' regulation of student learning. Our work addresses this interpretability issue with CNNs to provide useful features for student modeling applications that utilize CNNs.

One particular application that requires understanding the patterns learned by a model (or any method) is sequential pattern mining. The typical approaches for sequential pattern mining is to identify a set of elements that frequently co-occur; in sequence data, that corresponds to finding a set of events, items, locations, etc. that often happen sequentially in the data [1, 3, 5, 6, 7, 8, 13, 14, 23, 25]. However, those methods of pattern discovery suffer from problems like pattern explosion [21], which occurs when the number of frequent patterns is myriad and the importance (or usefulness) of patterns is uncertain. Consequently, the large number of patterns and prioritization of common patterns can be especially problematic in educational data where low-support patterns may be of interest (e.g., when examining uncommon patterns specific to students from underrepresented demographic groups [20]). In addition, existing methods do not consider the context of a pattern. We aim to train convolutional neural networks that have inherently interpretable features (i.e., discrete absence/presence of a specific student event, like watching a video or posting to a discussion forum) and enforce learning of patterns that predict context. We can thus interpret and utilize these patterns in downstream tasks in the same way that patterns mined via sequence mining methods are.

This paper aims to train a CNN with self-supervised learning to produce a model that can predict future actions based on sequences of events, thereby encouraging the model to learn *predictive* sequences (rather than frequent, unique, or other criteria). The features learned by self-supervised neural networks can be shared by various downstream tasks, as has been demonstrated in previous research. In this paper, we report results from a CNN trained to predict future student activities from past activities, and which thus captures event dependencies.

To show the effectiveness of the proposed method, we applied it two large datasets of student actions logged in learning environments, and utilized transfer learning to predict student outcomes with features derived from the discovered patterns. Specifically, after learning the patterns from student data, we generated feature representations from each

pattern and trained a supervised model to predict students' grade outcomes. We demonstrate that in several cases our results outperformed and were more stable than a typical sequential pattern mining method.

In summary, our contributions include two parts:

1. We trained interpretable CNN filters to explicitly learn patterns consisting of either mutually-exclusive (unique) or concurrent (co-occurring) elements (i.e., actions).

2. We evaluated the quality of patterns learned with our method in a transfer learning task involving prediction of students' outcomes in two datasets.

2. APPROACH

Our goal is to find frequent, predictive patterns with fixed length given sequences of actions done by students (or events, or items). Each step can contain either a single unique action or multiple concurrent actions (depending on the dataset), which can be regarded as sequential actions, events, activities, or other categorical values. In this section, we first explain the framework of the unique event pattern detector, which is the simplest case and perhaps the most widely-applicable. We then describe the solution for the multiple concurrent events pattern detector, as an illustration of how the unique-element approach can be generalized to other variations of the problem. We also describe a "warm-up" strategy, which is necessary to effectively train the pattern mining models. During the evaluation phase, we then derive features from the extracted patterns and apply them to a supervised student outcome prediction task as a measure of the quality of the patterns.

2.1 Unique Element Patterns Detector

We begin with the representation for each action and propose our pattern detection model for unique element sequence mining At the first stage, we use one-hot encoding to represent each action that was taken by students. After that, we use a one-dimensional CNN (i.e., convolving only over time), without bias weights, to extract patterns of action subsequences. We constrain the parameters of CNN filters to directly impose an interpretable, discrete structure on the weights. To predict future actions, we append a fully-connected layer and sigmoid function.

The crux of our approach to discovering interpretable patterns of specific actions is to force each row (corresponding to one step in a sequence) in the CNN filters to have only one parameter that is close to 1, while all others are close to 0. To achieve the desired weight structure, we applied regularization to CNN filter parameters as part of training.

In our method, the primary training objective is to minimize the binary cross-entropy loss for predicting future actions. To enforce discrete structure of the filters of CNN, we utilize regularization to force the sum of each row of the parameters of each CNN filter to 1, while most parameters are 0, thereby leaving only a single 1 corresponding to a single action. We split the approach into two steps. The first step is to ensure the sum of the parameters in each row is close to 1, by adding the loss:

$$L_{\text{r\_sum}} = \sum_{p=1}^{M}\sum_{n=1}^{k}\left(1 - \sum_{j=1}^{d} W_{pnj}^{3}\right)^{2} \tag{1}$$

where W refers to weight of convolutional neural network, d is the number possible actions (i.e., the size of each one-hot encoded vector), k is the number of sequence steps in each CNN kernel (i.e., the length of pattern to learn), and M is the number of filters in the CNN (i.e., the number of patterns to learn).

The second step is to encourage parameters to go toward 0 via row-wise L1 loss, leaving only one parameter close to 1 to minimize the $L_{\text{r\_sum}}$ row sum loss.

$$L_{\text{r\_l1}} = \sum_{p=1}^{M}\sum_{n=1}^{k}\sum_{j=1}^{d}|W_{pnj}| \tag{2}$$

Finally, we optimize the following joint objective function during training:

$$L = L_{\text{prediction}} + \alpha L_{\text{r\_sum}} + \beta L_{\text{r\_l1}} \tag{3}$$

where L_r is the whole structured regularization loss, and α and β are coefficients for each regularization part, included to balance the contrasting minimization objectives of $L_{\text{r\_sum}}$ and $L_{\text{r\_l1}}$.

2.2 Multiple Concurrent Elements Pattern Detector

The limitation of the unique action detector is that it can only handle situations where each step in the sequence contains exactly one action. In some circumstances, each step contains many actions or events, such as when a student does several activities logged with the same timestamp. We extended our approach to handle this condition, following the model proposed in the previous section with different constraints. Specially, we force each CNN kernel weight to be either close to 0 or close to 1, ignoring the sum of all weights and thus allowing multiple actions per step. To achieve this goal, the regularization loss for each parameter is minimized when the parameter is either 0 or 1.

We operationalized this regularization goal via the following quadratic equation:

$$L_{\text{r\_m}} = \sum_{p=1}^{M}\sum_{n=1}^{k}\sum_{j=1}^{d}|W_{pnj}^{2} - W_{pnj}| \tag{4}$$

Overall, the objective function is:

$$L = L_{\text{prediction}} + \gamma L_{\text{r\_m}} \tag{5}$$

In $L_{\text{r\_m}}$, γ serves as a weight we tuned to ensure that the structured regularization loss $L_{\text{r\_m}}$ has the desired effect on the CNN weights without over-emphasizing regularization relative to the prediction loss.

The prediction loss for our multiple concurrent elements example is to minimize binary cross-entropy with logits loss, though other loss functions could be applied.

2.3 Warm-up Period

Table 1: Performance comparison (Pearson's r correlation coefficient between predicted and actual student grades) of our method versus CM-SPAM on two datasets. The EPM dataset has grades for five learning sessions (labeled 2–6), while OULAD has grades for seven courses (labeled A–G). Results without warm-up and structured regularization are provided as points of comparison, though the CNN filters without structured regularization are not interpretable.

	EPM					OULAD						
	Course 2	3	4	5	6	Course A	B	C	D	E	F	G
CM-SPAM [5]	-.050	.603	.139	.134	.333	.318	.341	.341	.440	.381	.381	.510
Without warm-up	**-.032**	.729	.425	.055	.430	.324	.414	.514	.433	.456	.394	.544
Our approach	-.092	**.792**	.432	**.227**	**.450**	.330	**.461**	**.532**	**.500**	.498	**.433**	**.563**
Traditional CNN	-.199	.672	**.518**	.209	.375	**.365**	.416	.547	.511	**.506**	.422	.550

Frequently, models learned a local optimum where regularization losses were immediately optimized. To avoid getting stuck at local optima of the objective, we introduced a "warm-up" period to stabilize training [17]. In our experiment, we trained the model without regularization loss for five epochs. Subsequently, we linearly increased the coefficient of the regularization loss over the course of ten epochs.

2.4 Features for Transfer Learning

After learning the set of predictive patterns, we evaluated the utility of learned patterns for a subsequent prediction task (i.e., transfer learning with the learned patterns). We froze the weights of the CNN, then applied the network to generate pattern features for each student's sequence of actions. Note that each sequence is typically much longer than the number of steps in each CNN kernel. Thus, we aggregated filter activations for each pattern by applying basic statistical calculations, including sum, standard deviation, max, min, skew, kurtosis, and different quantiles (10%, 30%, 50%, 70%, 90%).

We then concatenated all of these aggregated values of all extracted patterns to create feature vectors. As a means to judge the quality of the pattern features, we predicted students' learning outcomes with a random forest regression model [4].

3. EXPERIMENTS

In this section, we first introduce the details of two datasets and a baseline pattern discovery algorithm, against which we compare our proposed method (Table 1). We use visualization to examine the learned patterns, and compare transfer learning predictions of student outcomes via Pearson correlations. Finally, we discuss the convergence of our method.

3.1 Datasets

We work on two public datasets that contain learning behaviors of students represented by actions from different courses.

Educational Process Mining (EPM). The EPM dataset [19] contains sequential records of 100 students' activities during 6 laboratory sessions (5 of which have outcome labels) of the digital design course at the University of Genoa. Actions were logged in sequential order, such that each row represented a unique action taken by a student. We describe activities included in EPM dataset, including their frequency, in the Appendix.

Open University Learning Analytics Dataset (OULAD). The OULAD dataset [12] contains data about courses, students, actions of students, and their interactions with a virtual learning environment (VLE; specifically, Moodle) for seven courses, which started from either February or October. We merged multiple semesters of the same course because the patterns in the same courses should be relatively (if not exactly) consistent. The detail of interaction events included in the dataset shown in the Appendix (we merged some infrequent interactions into *other* category because the frequency of occurrence of these interactions was rare).

3.2 Baseline Comparison Method

Typical sequential pattern mining algorithms include those like CM-SPAM, GSP [18], PrefixSpan [8], and SPADE [25]. We use CM-SPAM as a baseline method here because it can easily find patterns of a specific length, which allows fair comparison to our proposed method.

CM-SPAM [5] is a sequential pattern mining algorithm based on Sequential PAttern Mining (SPAM; [3]). SPAM is a depth-first sequential frequent pattern search algorithm. CM-SPAM prunes the SPAM search space to improve computational complexity. We focused on mining patterns with the highest support and matched the length of patterns in our method, selecting 25 of the highest-support patterns to compare against the 25 patterns learned by our method.

3.3 Experimental Setup

We optimized CNN models with Adam [10] for 50 epochs. We tuned hyperparameters including learning rate, loss coefficients, and warm-up duration, based only on results in OULAD course A, to avoid over-fitting hyperparameters to the other six OULAD courses or the EPM dataset. Hyperparameters related to the structure of input and the model architecture we left fixed. Specifically, we convolved CNN filters of length 3 over subsequences of current events of length 5, with stride length 1, and predicted the next 1 event. Models had 25 convolution filters (patterns to learn). We concatenated convolution filter outputs and used a fully-connected layer with sigmoid activation for predicting the next action. We found that the model worked best with the learning rate set to .001, after testing .01, .0075, .005, .0025, and .001.

Components of the structured regularization loss have notably different magnitudes for the unique action case, since the L1 loss component ($L_{\text{r\_l1}}$) is several times larger than

the filter row sum component ($L_{\text{r\_l1}}$). We thus applied a relatively large weight for $L_{\text{r\_l1}}$ and small weight for $L_{\text{r\_l1}}$ to balance regularization terms and achieve the desired weight structure. We tried different ratios of α/β, including 1, 10, 20, 30, 40, 50, 60, 70, 100, and 200. With $\alpha = 0.075$ and $\beta = 0.0075$, loss converged well. For the warm-up procedure, after testing 1 epochs, 5 epoch, and 10 epochs, we found the model converged best with $n_{total} = 5$ epochs.

For evaluating pattern utility via transfer learning with random forest regression, a higher correlation score (Pearson's r ranging from -1 to 1) represents patterns with higher utility for the downstream task.

3.4 Performance Analysis

Outcome prediction is a natural way to evaluate the utility of patterns [11]. We did so using the transfer learning approach described above, and split each course from each dataset into a train/test set at the student level with a ratio of 2:1 for evaluation.

3.4.1 Quantitative Results

The results of our method are shown in Table 1. Because students' learning actions contain meaningful sequential dependencies with each other, which patterns that happen frequently do not naturally capture. Consequently, CM-SPAM patterns were slightly less useful for inferring high level information (predicting student outcomes), as shown in Table 1. Additionally, instructors may be able to interpret the patterns extracted by our methods to intervene in future courses, given that the extracted patterns are few enough in number (25) to manually review and are related to outcomes. Generally, our approach outperformed traditional sequential pattern mining for almost all courses across the two datasets. Our approach improved the correlation of sequences with outcomes by an average of $r = .131$ on the EPM dataset, and was as good or better than CM-SPAM (r improved by .101) on the OULAD dataset. The result confirmed the usefulness of our predictive patterns derived via self-supervised learning.

3.4.2 Pattern Visualization and Analysis

We visualize the patterns extracted by our approach, demonstrate that they have the desired structure, and compare them with the traditional CNN filters.

To compare our method and typical CNN patterns, typical densely-distributed CNN weights lend little insight into the specific sequences of actions that activate filters (shown in Figure 1). These CNN patterns conflate selection of relevant input actions with weighting those patterns, which prevents their use as a sequence mining method. In addition, typical CNNs only extract patterns that correlate with student outcomes (in fully supervised applications). As a result, they do not necessarily learn dependencies among students actions; it remains to be seen whether our method applied in a fully-supervised model would produce notably different sequences of behaviors. Regardless, the patterns learned by our method are interpretable relative to typical CNN filter weights, which makes them straightforward to utilize for other downstream tasks even though they come from a self-supervised model. However, patterns with multiple concurrent actions are still less easily interpreted than patterns for unique actions given the possibility of many different concurrent actions. They are, however, still straightforward to transfer to downstream tasks.

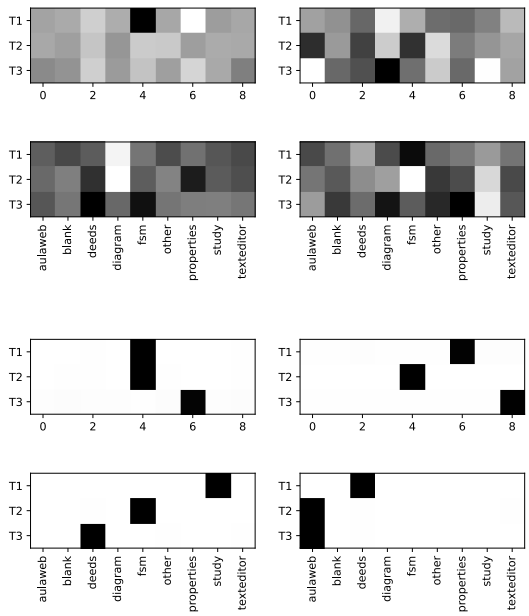

Figure 1: The bottom two rows are randomly-chosen example patterns extracted by our approach based on EPM course 2. The top two rows are traditional CNN filters.

4. CONCLUSION

In this paper, we presented a general self-supervised sequence mining algorithm that works for both sequences of individual actions and multiple concurrent actions. We mined sequential patterns by convolutional neural networks, and applied transfer learning to judge the quality of the extracted patterns for predicting student outcomes as an example downstream prediction task. Our results showed that the extracted patterns were indeed useful, as measured by the correlation between predictions and student outcomes.

We empirically demonstrated that the patterns extracted by our method have similar or higher utility for two prediction tasks than those extracted via a traditional frequent pattern mining algorithm, while the extracted patterns can still be easily interpreted. Furthermore, our approach deals with common pattern mining problems like pattern explosion by training a fixed number of convolutional filters, where filters are selected from the space of all possible filters via stochastic gradient descent.

In summary, our approach is a novel and interpretable way to extract predictive patterns of actions from sequential data.

5. REFERENCES

[1] R. Agrawal, R. Srikant, et al. Fast algorithms for mining association rules. In *Proceedings 20th International conference Very Large Data Bases,*

VLDB, volume 1215, pages 487–499, 1994.

[2] Ş. Aydoğdu. A new student modeling technique with convolutional neural networks: Learnerprints. *Journal of Educational Computing Research*, page 0735633120969216, 2020.

[3] J. Ayres, J. Flannick, J. Gehrke, and T. Yiu. Sequential pattern mining using a bitmap representation. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 429–435, 2002.

[4] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[5] P. Fournier-Viger, A. Gomariz, M. Campos, and R. Thomas. Fast vertical mining of sequential patterns using co-occurrence information. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 40–52. Springer, 2014.

[6] M. N. Garofalakis, R. Rastogi, and K. Shim. Spirit: Sequential pattern mining with regular expression constraints. In *VLDB*, volume 99, pages 7–10, 1999.

[7] J. Han, J. Pei, B. Mortazavi-Asl, Q. Chen, U. Dayal, and M.-C. Hsu. Freespan: frequent pattern-projected sequential pattern mining. In *Proceedings of the sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 355–359, 2000.

[8] J. Han, J. Pei, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M. Hsu. Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth. In *Proceedings of the 17th International Conference on Data Engineering*, pages 215–224. Citeseer, 2001.

[9] T. Hu, G. Sun, and Z. Xu. Assessing student contributions in wiki-based collaborative writing system. In *Proceedings of The 13th International Conference on Educational Data Mining (EDM 2020)*, pages 615–619, 2020.

[10] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[11] S. Klingler, R. Wampfler, T. Käser, B. Solenthaler, and M. Gross. Efficient feature embeddings for student classification with variational auto-encoders. *International Educational Data Mining Society*, 2017.

[12] J. Kuzilek, M. Hlosta, and Z. Zdrahal. Open university learning analytics dataset. *Scientific Data*, 4:170171, 2017.

[13] M.-Y. Lin and S.-Y. Lee. Fast discovery of sequential patterns by memory indexing. In *International Conference on Data Warehousing and Knowledge Discovery*, pages 150–160. Springer, 2002.

[14] H. Pinto, J. Han, J. Pei, K. Wang, Q. Chen, and U. Dayal. Multi-dimensional sequential pattern mining. In *Proceedings of the Tenth International Conference on Information and Knowledge Management*, pages 81–88, 2001.

[15] S. Shen, Q. Liu, E. Chen, H. Wu, Z. Huang, W. Zhao, Y. Su, H. Ma, and S. Wang. Convolutional knowledge tracing: Modeling individualization in student learning process. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1857–1860, 2020.

[16] X. Shen, B. Yi, Z. Zhang, J. Shu, and H. Liu. Automatic recommendation technology for learning resources with convolutional neural network. In *2016 International Symposium on Educational Technology (ISET)*, pages 30–34. IEEE, 2016.

[17] C. K. Sønderby, T. Raiko, L. Maaløe, S. K. Sønderby, and O. Winther. Ladder variational autoencoders. In *Advances in Neural Information Processing Systems*, pages 3738–3746, 2016.

[18] R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, pages 1–12, 1996.

[19] M. Vahdat, L. Oneto, D. Anguita, M. Funk, and M. Rauterberg. A learning analytics approach to correlate the academic achievements of students with interaction data from an educational simulator. In *Design for Teaching and Learning in a Networked World*, pages 352–366. Springer, 2015.

[20] H. Valdiviejas and N. Bosch. Using association rule mining to uncover rarely occurring relationships in two university online stem courses: A comparative analysis. In *Proceedings of The 13th International Conference on Educational Data Mining (EDM 2020)*, pages 686–690, 2020.

[21] M. Van Leeuwen. Interactive data exploration using pattern mining. In *Interactive knowledge discovery and data mining in biomedical informatics*, pages 169–182. Springer, 2014.

[22] R. Wampfler, A. Emch, B. Solenthaler, and M. Gross. Image reconstruction of tablet front camera recordings in educational settings. *International Educational Data Mining Society*, 2020.

[23] J. Wang and J. Han. Bide: Efficient mining of frequent closed sequences. In *Proceedings 20th International Conference on Data Engineering*, pages 79–90. IEEE, 2004.

[24] Y. Xiao, G. Zingle, Q. Jia, H. R. Shah, Y. Zhang, T. Li, M. Karovaliya, W. Zhao, Y. Song, J. Ji, et al. Detecting problem statements in peer assessments. *arXiv preprint arXiv:2006.04532*, 2020.

[25] M. J. Zaki. Spade: An efficient algorithm for mining frequent sequences. *Machine Learning*, 42(1-2):31–60, 2001.

6. APPENDIX: DATASET DETAILS

Table 2: Description of actions in the OULAD dataset. Infrequent actions were grouped together into an *other* category, with the exception of *transfer* given that it is one of the most semantically important, along with *register* and *unregister*.

Action	Description	Frequency
homepage	Visit the main course page	1,735,226
gap	One or more consecutive days with no action	860,356
oucontent	View course content page	829,476
forumng	Discussion forum usage	822,895
subpage	Manage/view course activities on a page other than the homepage	804,577
resource	Download a document from the course	399,961
url	Click a link to an external site	314,240
quiz	Take a quiz	211,497
exam	Take an assessment	160,498
ouwiki	Access the course wiki	89,406
register	Register for the course	32,548
unregister	Drop the course	10,072
transfer	Transfer grade from previous session (semester)	526
	Infrequent activities grouped together as "other"	
page	Non-interactive information page	47,549
oucollaborate	Audio/video conferencing	47,334
externalquiz	Externally-hosted quiz	41,642
glossary	View course glossary	17,258
questionnaire	Access survey form	15,109
ouelluminate	Audio-only conferencing	11,384
dualpane	Side-by-side view of instructions and related content	9,256
dataplus	Interact with a toy SQLite database	6,818
htmlactivity	Interactive HTML page	6,016
folder	View folder containing related activities	4,678
sharedsubpage	View page shared from another course	148
repeatactivity	Activities repeated from earlier in the course	3

Table 3: Description of activities in the EPM dataset. Internal activity names from the EPM dataset are provided to enable unambiguous matching to the original data.

Action	Description	Frequency
texteditor	Use a text editor	42,431
deeds	Other DEEDS (Digital Electronics Education and Design Suite) activities	38,372
other	Not viewing any pages described above (mostly off-task activities)	33,602
blank	Title of visited page is not recorded	24,303
study	View exercises or materials related to courses	22,261
diagram	Use a "simulation timing diagram" to test a solution	20,815
fsm	Use a finite state machine (FSM) simulator	20,596
properties	Set parameters of a simulation or design	19,677
aulaweb	Visit learning management system	8,261