Early Dropout Prediction with Neural Co-embeddings

Unknown author for double blind evaluation

Abstract

We present an approach to dropout prediction in Massive Online Courses (MOOCS) that relies on a neural model of student behavior. The evaluation is focused on predicting the dropout weekly, using partial information, to simulate a more realistic scenario.

We propose to obtain a joint representation (a co-embedding) of students and course components with a recurrent neural network (RNN) trained with logs of student activity. A joint representation is more adequate than disjoint representations because they elicit insights on the interactions between students and contents. Such insights are useful for early prediction, when less information is available. This approach does not require manual labeling of the data, which makes it less prone to theoretical bias, more portable and less costly to develop. Results indicate that a joint embedding improves the performance for datasets with less students.

Introduction and Motivation

The ultimate goal of education is to improve the learning process, and in particular Educational Data Mining (EDM) contributes to such objective through the analysis of data. Human learning is a complex process, with hidden, unknown causes governing the behavior of students and the success or failure of courses. This turns the analysis of educational data into a search for these hidden causes that explain or predict a certain phenomenon of interest.

The field of EDM has been growing steadily in the past decade, slowly including more machine learning and data science approaches. The increased development of Massive Open Online Courses (MOOCs) and the new availability of Intelligent Tutoring Systems (ITS) allow researches to gather large amounts of data.

Dropout Detection

Dropout detection is one of the most studied tasks in EDM, motivated by rates of students prematurely leaving MOOCs as high as 93% (Yang et al. 2013). Some lines of work focus on the prediction of student behavior at the end of the course, while others seek to prevent dropout and focus on the prediction of actions in the near future (Whitehill et al. 2017).

The KDDCup 2015 competition [1] considered as dropout the absence of student activity in the 10 days following the end of the course. It was not clear, however, the nature of such interaction or whether a posterior activity was present or not. There are more possible definitions of dropout, as proposed by (Fei and Yeung 2015) , but most of them can only be assessed after the course has ended. In consequence, it is hard to train a model while the course is running, when an intervention is still possible.

From a more general point of view, educational content is so rich in different kinds of information that it becomes problematic to discover patterns and systematize them in a uniform manner. Even courses created in a single platform can be very different, with disjoint sets of lessons, designed for different levels of engagement, among others. These phenomena are aggravated between different platforms, which can collect different signals or provide different tools. It is hard to evaluate a dropout prediction model across several platforms, or develop a generalizable framework.

Furthermore, in MOOCs students come from very different backgrounds and have different expectations as well. (Kizilcec, Piech, and Schneider 2013) analyze the background, demographics and common learning paths of students in MOOCs.

Traditional MOOC platforms generate records of student activity with a very low level of granularity. It is a challenge to interpret these logs in tasks like dropout prediction, which involve high-level concepts like skills or engagement.

Contribution

Neural models (or deep models) have proven useful to find suitable representations for complex problems in several areas: from modeling meaning in natural language to detecting nonconcrete concepts in image classification. The multiple layers in a deep architecture build a number of internal representations of the input data, optimized for a certain prediction task. These internal representations are liable to capture underlying causes of behaviour like the diversity in student background and interests.

The main contribution of this work is the evaluation of neural classifiers used for early dropout prevention, simulat-

[1] https://biendata.com/competition/kddcup2015/

ing a realistic environment. The data is divided into periods of one week and the models trained to predict the presence of any activity on the following week. This approach produces models that can be applied as the course progresses, and thus allow for timely dropout detection and intervention. The difficulty of this approach is the lack on initial information to train the model. Neural models need large amounts of examples to avoid falling into local minima.

To tackle this challenge, we propose to model student state with course elements using embeddings on a shared space. Student embeddings capture the factors relevant to the engagement level, while course element embeddings optimize the representation of materials for dropout prediction. The use of a joint embedding space helps the classifier to model better the relations between the sequence of visited course elements and student engagement, adding more information to scenarios with little student actions recorded.

Additionally, given the neural structure of the embeddings, they can be obtained from low-level data, using only the identifiers of the visited course elements. As a result, the method is flexible enough to be easily ported to different frameworks and course types, because low-level data is quite comparable across courses. It also has the advantage that they do not require that experts previously annotate the data in each course, and can be thus adapted to MOOCs with lower resources.

Experimental results indicate that models with embeddings and co-embeddings outperform a basic recurrent architecture, and are particularly helpful for the detection of dropout on the first week. In particular, they increase performance in smaller courses, which suffer more acutely from lack of generalization.

Relevant work

As mentioned in the previous section, many studies have been carried out for dropout prediction using deep learning. For the task of early prediction, it is possible to train a model using historical data and use it to detect near dropout cases in a new, current course. However, the diversity between MOOCs hinders the possibility of transfer learning between different courses. As pointed out by (Whitehill et al. 2017) and (Xing et al. 2016), many dropout predictors are trained and evaluated using the full history of students' actions on the same course.

An example is the work of (Fei and Yeung 2015), where the authors use a neural model and report an increase in performance using the data aggregated in weekly periods instead of the full sequence of interactions.

It is also worth noticing that the factors involved in dropout prediction are not thoroughly understood. For example in (Kizilcec, Piech, and Schneider 2013) an analysis is done on the difference of background and expectations from students, which leads to diverse levels of engagement.

Deep learning architectures have been proven very helpful in other EDM tasks like Deep Knowledge Tracing (DKT) (Piech et al. 2015), where the level of knowledge of students is modeled using the hidden state of a recurrent neural network. This representation is obtained by optimizing the model to predict if the student will be able to solve the next exercise presented by a tutoring system. In contrast with other non-neural methods, it does not depend on manually assigned skills. The work of (Tang, Peterson, and Pardos 2016) models students also using RNNs trained to predict the next action to be performed by the student.

Co-embeddings have been successfully used before in MOOC environments. (Reddy, Labutov, and Joachims 2016) propose an embedded representations of students in a "latent skill space" that can be interpreted as the knowledge level of the student on each skill. Along with them, they find embeddings for course elements in the same skill space. As this approach is based on MOOC-like course structures, it takes into account both graded (assessments) and non graded (lessons) types of course elements.

Neural co-embeddings for student states and course elements

As presented in previous sections, our aim is to obtain a joint representation of students and course elements that highlights the major factors involved in the human learning experience. The goal is to find the embedding functions φ_E and φ_S that project course elements and students from their representation in log data into a shared space, respectively. This method has been proposed and evaluated for other tasks in (Teruel and Alonso Alemany 2018). In the shared space each course element is represented as a vector. Students are represented as a sequence of vectors, each of them corresponding to a point of time. Individual vectors represents the state of the student after each interaction with a course element.

To optimize the co-embeddings, we propose a recurrent neural architecture, shown in Figure 1. The base of this architecture is a Recurrent Neural Network (RNN), which is designed to collect the relevant information of the inputs previously seen. The network is trained with the ids of the course elements seen by students.

The specific update equations for a basic RNN are the following:

$$y^{(t)} = \sigma(W^{(hy)}h^{(t)} + b^{(y)})$$
$$h_t = tanh(W^{(xh)}x^{(t)} + W^{(hh)}h^{(t-1)} + b^{(h)}) \quad (1)$$

Where the upper index t indicates the time step. $y^{(t)}$ is the output of the network, given the input $x^{(t)}$. h_t is the hidden state, which also represents the student embedding. The matrices W and biases b are the network's parameters.

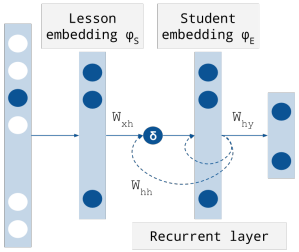
To integrate the embedding of course elements into the RNN classifier, we add a new embedding layer between the input layer and the recurrent layer. However, ensure the students and for course elements are effectively in the same space, we use as input to the recurrent layer a point-wise combination between both embeddings.

After introducing these changes, Equation 1 is rewritten as:

$$h_t = tanh(W^{(xh)}\delta(\varphi_E(x^{(t)}), h^{(t-1)}) + W^{(hh)}h^{(t-1)} + b^{(h)})$$

Student and course element has been combined before by (Reddy, Labutov, and Joachims 2016). However, the authors use only the length of projection of the student em-

Figure 1: Co-embedded recurrent architecture



bedding over the lesson embedding, while we use the pointwise function δ. This allows us to discriminate the influence of each dimension in the resulting state, not compensating a small value on a dimension with a large value in another.

We propose several δ functions, some examples are: $\delta(x, y) = (x - y)^2$, $\delta(x, y) = |(x - y)|$ and $\delta(x, y) = norm(x - y, 0, std)$ (all operations are pointwise). In the last function, $norm$ refers to the probability of the vector following a normal distribution

Experimental setting

The dataset used for the experimental evaluation is the KDDCup 2015 competition. It was provided by XuetangX, a Chinese MOOC learning platform initiated by Tsinghua University, a partner of EdX.

Although the information is no longer available in the competition website, this dataset was, to our knowledge, one of the few freely available big datasets of detailed logging in a MOOC environment. The data provided logs of events like access to video content, resolution of a problem, etc., for 39 different courses. Events are timestamped and identified with the corresponding student and course.

Looking for a more realistic scenario, we have performed a **period-wise** prediction, trying to emulate the conditions under which an early detection system could be used for dropout prevention. We divide each course into periods of 7 days. After this, we take as training instances the sequence of interactions up to the end on the period. We assign each instance a label representing whether the student has activity in the next period. In this dataset, all courses have a span of 4 weeks, allowing to evaluate 3 periods in period-wise prediction. The original definition of dropout classified 79% of all students as dropouts, and he distribution of dropouts is similar across courses moving between 70% and 90%. Using the period-wise defition of dropout, we see a higher concentration of dropouts in latter periods, with a mean of 0.67 for the first period, and a mean of 0.81 for the last two.

One model was trained for each course, and we report the results of the model with better AUC. We have found that courses with different numbers of students have very different results, therefore we decided to distinguish results in three different segments of courses: 5 big courses with more than 6000 training students, 9 medium courses with between 5000 and 2000 students and 24 small courses with less than 2000 students.

The entire student sequences of actions were divided in three portions, training (70%), testing (20%) and validation (10%). The performances reported are obtained by applying the trained model over the testing dataset, after the best hyperparameters have been chosen using the validation dataset.

The neural architectures we tested are all based on RNNs. The simplest one (**LSTM**) has a single layer of LSTM cells. The input for this model is the one-hot encoding representations of the course element id. The output of the recurrent layer is connected to a dropout layer, and later to a regular dense layer with sigmoid activation and L2 regularization. The output layer is composed of two neurons with softmax activation, one for each class. In the second model (**E-LSTM**) the input layer is replaced by an embedding layer. Since students and course elements are not forced to share the same space, we call this approach *disjoint embeddings*. The final model (**CoE-LSTM**) is the one described in section , implementing the proposed co-embeddings.

The algorithm used to optimize a recurrent neural architecture is called Back Propagation Through Time (BPTT). However, propagating the gradients over very long sequences of time can produce vanishing gradients. LSTM networks are designed to avoid vanishing gradients, but in practice they also have a limited propagation point. A technique used to overcome this problem is truncating the gradients after certain amount of steps, leading to a Truncated BPTT (TBPTT). However, as TBPTT decreased the performance significantly, we used only the last 100 steps of the sequences to train every model.

The hyperparamenters of the networks optimized included embedding size (20, 50, 100 and 200), hidden layer size (20, 50, 100 and 200), and dropout ratio (0, 0.2, 0.3 and 0.5). The optimizer used is the Adam implementation of Tensorflow with a learning rate of 0.001.

The metric used is the Area Under the ROC Curve (AUC), the reference metric in the KDDCup competence. It evaluates the performance of binary probabilistic classifiers in different thresholds, measuring also the difference with a random baseline.

Results

In Figure 3 we show the results for period-wise prediction, showing the AUC of the three classifier types on all courses, divided by period. We can observe that embedded models outperform LSTM models in all periods, thus the generalization provided by embeddings is useful for this task as well. The median of performance values in embeddings and co-embeddings are close in the two first periods. However, the difference in the quartiles indicates that co-embeddings are in fact outperforming disjoint representations. The presence of fewer outliers and lower dispersion in general of co-embedded models indicate the classifiers generalize better with respect to different dataset. Collectively, these results suggest joint models are more adequate to represent sequences with fewer interactions, and they are a good choice to apply dropout prevention in on-going courses.

When we disaggregate courses by size, we can see the difference between co-embedded models is more significant in smaller courses, even if the general performance

Figure 2: AUC for period-wise prediction, divided by course size

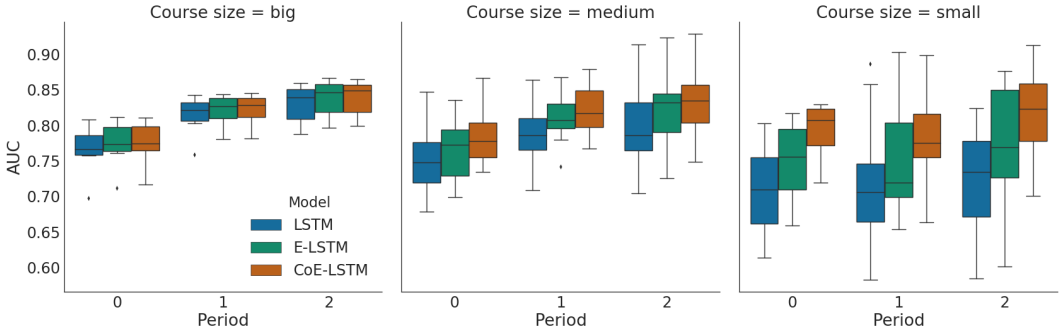
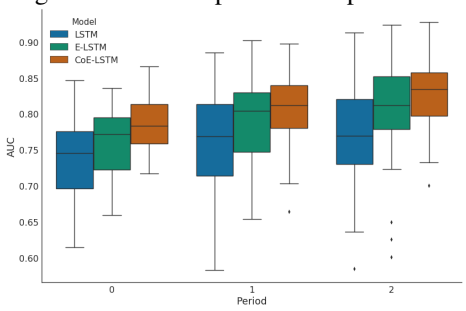


Figure 3: AUC for period-wise prediction



in such courses is worse than for bigger courses. In Figure 2 we can see that for the biggest courses all classifiers perform indistinguishably, with small variations, hence the smaller spread of the boxplot and close medians. However, for smaller courses, where data is more scarce and performance is worse, co-embeddings seem to provide useful generalizations over the low-level data.

Conclusions

We have reformulated the dropout task into a period-wise prediction scenario, more analogous to the scenario where prevention policies could be deployed. We have assessed the impact of joint embeddings of course elements and students for this evaluation setting.

Results indicate that co-embeddings are able to capture the latent causes involved in dropout, outperforming disjoint and not-embedded representations. Gaining insight into the results, we show that performance is better in courses with less students.

We have obtained promising results from the performance point of view, but there is still work to do on the effectiveness of the joint representation for interpretation of human learning.

References

Fei, M., and Yeung, D.-Y. 2015. Temporal models for predicting student dropout in massive open online courses. In *Data Mining Workshop (ICDMW), 2015 IEEE International Conference on Data Mining*, 256–263. IEEE.

Kizilcec, R. F.; Piech, C.; and Schneider, E. 2013. Deconstructing disengagement: analyzing learner subpopulations in massive open online courses. In *Proceedings of the third international conference on learning analytics and knowledge*, 170–179. ACM.

Piech, C.; Bassen, J.; Huang, J.; Ganguli, S.; Sahami, M.; Guibas, L.; and Sohl-Dickstein, J. 2015. Deep knowledge tracing. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, NIPS'15, 505–513. Cambridge, MA, USA: MIT Press.

Reddy, S.; Labutov, I.; and Joachims, T. 2016. Learning student and content embeddings for personalized lesson sequence recommendation. In *Proceedings of the Third (2016) ACM Conference on Learning @ Scale*, L@S '16, 93–96. New York, NY, USA: ACM.

Tang, S.; Peterson, J. C.; and Pardos, Z. A. 2016. Deep neural networks and how they apply to sequential education data. In *Proceedings of the Third (2016) ACM Conference on Learning @ Scale*, L@S '16, 321–324. New York, NY, USA: ACM.

Teruel, M., and Alonso Alemany, L. 2018. Co-embeddings for student modeling in virtual learning environments. In *Proceedings of the 26th Conference on User Modeling, Adaptation and Personalization*, UMAP '18, 73–80. New York, NY, USA: ACM.

Whitehill, J.; Mohan, K.; Seaton, D.; Rosen, Y.; and Tingley, D. 2017. Mooc dropout prediction: How to measure accuracy? In *Proceedings of the Fourth (2017) ACM Conference on Learning @ Scale*, L@S '17, 161–164. New York, NY, USA: ACM.

Xing, W.; Chen, X.; Stein, J.; and Marcinkowski, M. 2016. Temporal predication of dropouts in moocs: Reaching the low hanging fruit through stacking generalization. *Computers in Human Behavior* 58:119–129.

Yang, D.; Sinha, T.; Adamson, D.; and Rose, C. P. 2013. Turn on, tune in, drop out: Anticipating student dropouts in massive open online courses. In *Proceedings of the 2013 NIPS Data-Driven Education Workshop*, volume 10, 13–20.