Transfer Learning for Predictive Models in Massive Open Online Courses

Sebastien Boyer and Kalyan Veeramachaneni(✉)

Computer Science and Artificial Intelligence Laboratory,
Massachusetts Institute of Technology, Cambridge, USA
{sebboyer,kalyan}@csail.mit.edu

Abstract. Data recorded while learners are interacting with Massive Open Online Courses (MOOC) platforms provide a unique opportunity to build predictive models that can help anticipate future behaviors and develop interventions. But since most of the useful predictive problems are defined for a real-time framework, using knowledge drawn from the past courses becomes crucial. To address this challenge, we designed a set of processes that take advantage of knowledge from both previous courses and previous weeks of the same course to make real time predictions on learners behavior. In particular, we evaluate multiple transfer learning methods. In this article, we present our results for the stopout prediction problem (predicting which learners are likely to stop engaging in the course). We believe this paper is a first step towards addressing the need of transferring knowledge across courses.

1 Introduction

Data recorded while learners are interacting with the Massive Open Online Courses (MOOC) platform provide a unique opportunity to learn about the efficacy of the different resources, build predictive models that can help develop interventions and propose/recommend strategies for the learner. Consider for example a model built to predict *stopout*, that is, how likely is the learner to stop engaging with the course in the coming weeks. One can learn the model retrospectively on data generated from a finished course by following a typical data mining procedure: splitting the data into *test-train*, learning the model and tuning its parameters *via* cross validation using the *training* data, and testing the models accuracy on *test* data which is a proxy for how model may perform on unseen data.

In this paper, we focus on operationalization of the predictive models for real time use. We raise a fundamental question: whether models *trained* on previous courses would perform well on a new course (perhaps a subsequent offering of the same course). This question is of utmost importance in MOOCs due to the following observations:

Offerings could be subtly different: Due to the very nascent nature of the online learning platforms, many aspects of the course evolve. Learners are thus placed in a different environment each time, so they may interact and behave differently.

C. Conati et al. (Eds.): AIED 2015, LNAI 9112, pp. 54–63, 2015.
DOI: 10.1007/978-3-319-19773-9_6

Offerings have a different learner/instructor population: Subsequent offeringsof a course have a different population of students, teaching assistants, and in some cases, different instructors.

Some variables may not transfer: Some of the variables developed for one offering may not exist in the subsequent offering. For example, if a variable is the time spent on a specific tutorial, the variable may not be computed if the tutorial is not offered in subsequent offering.

To develop methods for operationalization of models for real time predictions, in this paper we study three different offerings of a course offered by MIT via edX called *Circuits and Electronics*. We formulate two versions of the prediction problem. One version of the problem, in a very limited number of prediction scenarios, allows us to use data gathered in an on-going course to make predictions in the same course. We call this *in-situ* learning. In the second version we employ multiple transfer learning methods and examine their performance. In our models we rely on a set of variables that we consider are universal, and employ techniques to overcome the differences in their ranges.

The paper is organized as follows. In next section we present the datasets we are working with, and the features/variables we defined and extracted. In Section 3 we define the *stopout* prediction problem and multiple versions of the problem formulation. In Section 4 we present different transfer learning approaches we employ. Section 5 presents the results. Section 6 presents the related work in this area. Section 7 concludes.

2 Dataset

In a MOOC every mouse click learners make on the course website is recorded, their submissions are collected, and their interactions on forums are stored. In this paper, our data is from three consecutive offerings of an MITx course called *6.002x : Circuits and Electronics* offered *via* edX platform. We chose three offerings of this course as an ideal test case for transfer learning since we expect that different offerings of the same course may have statistically similar learner-behavior data. We name the offerings as A (offered in spring 2012), B (offered in fall 2012) and C (offered in spring 2013). The number of learners who registered/certified in these three offerings were 154753/7157, 51394/2987 and 29,050/1099 respectively.

Per Learner Longitudinal Variables. Even though each course could be different in terms of content, student population and global environment, the data gathered during the student's interaction with the platform allows us to extract a common set of longitudinal variables as shown in table 1 (more details about these features are presented in [12]).

3 Stopout Prediction in MOOCs

We consider the problem of predicting *stopout* (a.k.a *dropout*) for learners in MOOCs. The prediction problem is described as: considering a course of duration

Table 1. Features derived per learner per week for the above-mentioned courses

x_1	Total time spent on all resources
x_2	Number of distinct problems attempted
x_3	Number of submissions
x_4	Number of distinct correct problems
x_5	Average number of submissions per problem
x_6	Ratio of total time spent to number of distinct correct problems
x_7	Ratio of number of problems attempted to number of distinct correct problems
x_8	Duration of longest observed event
x_9	Total time spent on lecture resources
x_{10}	Total time spent on book resources
x_{11}	Total time spent on wiki resources
x_{12}	Number of correct submissions
x_{13}	Percentage of the total submissions that were correct
x_{14}.	Average time between a problem submission and problem due date

k weeks, given a set of longitudinal observations (*covariates*) that describe the learner *behavior* and *performance* in a course up until a week i predict whether or not the learner will persist in week $i + j$ to k where $j \in \{1 \ldots k - i\}$ is the *lead* [10]. A learner is said to *persist* in week i if s/he attempts at least *one* problem presented in the course during the week. In the context of MOOC's, making real time predictions about *stopout* would give a tremendous opportunity to make interventions, collect surveys and improve course outcomes.

3.1 Problem Formulation and Learning Possibilities

Given the learner interactions data up until week i, we formulate two different types of prediction problems. These different formulations have implications on how training data could be assembled and how model learning could incorporate information from a previous course.

Formulations

- **Entire history**: In this formulation, we use all the information available regarding the learner up until week i for making predictions beyond week i.
- **Moving window**: In this formulation, we use a fixed amount of historical information (parameterized by `window size`) of the learner to make predictions. That is, if the `window size` is set to 2 then for any week i we only use information from weeks $i - 1$ and $i - 2$.

Learning

- **In-situ learning**: In-situ learning attempts to learn a predictive model from the data from the on-going course itself. To be able to do so we have to assemble data corresponding to learners that *stopped* out (negative exemplars) along with learners who are *persisting* (positive exemplars). This is only possible for the moving window formulation. Under the formulation,

when the course is at week i and the `window size` is w, *lead* is j, and $w + j < i$, we can assemble training examples by following a sliding window protocol over the data up until i.
- **Transfer learning**: Through transfer learning we attempt to transfer information (training data samples or models) from a previous course to establish a predictive model for an ongoing course.

Performance Metric: We note that the positive examples in our dataset (learners that don't *stopout*) only represent around 10% of the total amount of learners. A simple baseline could lead to very high accuracy (e.g. predicting that every learner dropped out for all the test samples gives an accuracy of nearly 90%). We use Area Under the Curve as a metric to capture and compare the discriminatory performance of our models.

4 Transfer Learning

Notationally we call *source offering* S (for past source offering) and *target offering* T (for current on-going course). n_S and n_T respectively are the number of samples in the source and target offering and $D_S = \{x_{Si}, y_{Si}\}_{i=1}^{n_s}$ represents the data from the source. We distinguish two main scenarios for transfer learning as defined in [8] and [13]: **inductive transfer** learning where some labels are available for the target course given by $D_T = \{x_{Ti}, y_{Ti}\}_{i=1}^{n_t}$ and **transductive transfer** where no labels are available for the target course data, given by $D_T = \{x_{Ti}\}_{i=1}^{n_t}$.

4.1 A Naive Approach

When using samples from a previous course to help predict in a new course, we first wonder if the two tasks (predicting in the first course and predicting in the second course) are similar enough so that applying a model learnt on the first course to the second one would give satisfying results. To answer this question we train a logistic regression model with optimized ridge regression parameter on S and apply it to T. We call this *naive* transfer method.

4.2 Inductive Learning Approach

Multi-task Learning Method. In the multi-task learning method (MT)[2], two sets of weights are learnt for samples from source and target. The weights are coupled by having a common component that is shared between a weight for a covariate from source and target. This sharing is represented by:

$$\begin{cases} w_S = v_S + c_0 \\ w_T = v_T + c_0 \end{cases}$$

where $v_S, v_T \in \Re^d$, $d = m + 1$ for m covariates. From a logistic regression perspective, this method requires learning two coupled weight vectors, while regularizing the two components separately:

$$(w_S^*, w_T^*) = \arg \min_{v_S, v_T, w_0} \sum_{i=1:n_S} l(x_{Si}, y_{Si}, w_S) +$$

$$\sum_{i=1:n_T} l(x_{Ti}, y_{Ti}, w_T) + \frac{\lambda_1}{2}(||v_s||^2 + ||v_T||^2) + \lambda_2 ||c_0||^2$$

where $l(x, y, w) = \log(\frac{1}{1+\exp(-y(w^0 + x^T.w))})$. Regularization allows us to set the degree of similarity between courses via parameter $\frac{\lambda_2}{\lambda_1}$ stands for relative significance of the independent part over the common part of the weights. For small values of $\frac{\lambda_2}{\lambda_1}$ we expect the two models to have very common weight vectors (regularization constraints are higher on the distinct part v_S and v_T than on the common part c_0). In our experiments we set $\lambda_1 = 0.2$ and $\lambda_2 = 0.8$.

Logistic Regression with Prior Method. In MT method, the regularization parameters impose the same common/particular penalization ratio to all the components of the weight vector that correspond to different covariates. However, in most cases we would like to differentiate between covariates as their importance is likely to vary between source and the target course.

Hence we resort to a method that uses Prior distribution on the weights of the target model [3]. The Logistic Regression with Prior method, PM, first estimates the prior distribution (assumed to be Gaussian) on the weights by splitting the data from source course into $D = 10$ sub-samples (without replacement). For each sub-sample, we fit the usual logistic regression classifier using ridge regularization. We obtain D sets of weights $\{\{w_i^k\}_{i=1:d}\}_{k=1:D}$ and use them to compute our prior belief on the weights we expect to derive from any new course as follows:

$$\begin{cases} \mu_j = \dfrac{1}{D} \displaystyle\sum_{k=1:D} w_j^k \quad \text{for } j = 1:d \\ \sigma_j = \sqrt{\dfrac{1}{D-1} \displaystyle\sum_{k=1:D} (w_j^k - \mu_j)^2} \quad \text{for } j = 1:d \end{cases}$$

When building a model for the target domain, in this formulation, the usual logistic regression cost function$NLL(w_0, w) = \sum_{i=1:N} l(x_{Ti}, y_{Ti}, w_T) + \frac{1}{2}\lambda \sum_{j=1:d} w_j^2$ becomes $NLL(w_0, w) = \sum_{i=1:N} l(x_{Ti}, y_{Ti}, w_T) + \frac{1}{2}\lambda \sum_{j=0:d} \frac{(w_j - \mu_j)^2}{\sigma_j^2}$, where $\mu = [\mu_1, \ldots, \mu_d], \sigma = [\sigma_1, \ldots, \sigma_d]$ and $l(x_i, y_i, w) = \log(\frac{1}{1+\exp(-y_i(w^0 + x_i^T.w))})$ is the log-likelihood of the i *th* data point for a multivariate gaussian prior on the distributions. The effect of such strategy is to allow higher flexibility (less constraints) for weights which vary significantly across source sub-domains and constrain more the weights whose variation across source sub-domains are smaller. In our experiments, we set $\lambda = 1$.

4.3 Transductive Learning Approach

In this section we focus on the scenario where no labeled samples are available in the target course. This is the case when considering a *entire history* setting (no labeled samples from the ongoing course are available because the week of the prediction is in the future). To transfer the model from source to target, we follow our belief that the covariates from the two courses may overlap to a significant degree and we use a sample correction bias [5]. This importance sampling, or IS, method is equivalent to assuming that the covariates are drawn from a common distribution and that their difference comes from a selection bias during the sampling process (out of this general common distribution). In order to correct this sample selection bias, the idea is to give more weight to the learners in the source course that are "close" to the learners in the target course. Doing so, the classification algorithm takes advantage of the similar learners of the source course and barely considers the significantly different ones. For each target sample we predict: $\hat{y}_{Ti} = \arg \max_{y \in \{+1,-1\}} l(x_{Ti}, y, w^*)$. The weights are estimated from the source data by optimizing:

$$\text{where } w^* = \arg\min_{w} \sum_{i=1:n_S} \beta_i \, l(x_{Si}, y_{Si}, w)$$

Note that each learner's data is reweighted using a parameter β_i in the log likelihood objective function. Finding the optimal β_i for such a reweighting procedure would be straigthforward if we knew the two distributions from which the source and the target learners data are drawn. To find these in practice, we use a gaussian kernel $k(x, z) = \exp(-\frac{||x-z||^2}{\sigma^2})$ $\forall x, z \in S \cup T$ to measure the distance between data points. In our experiments $\sigma = 1$. We then compute for each source data point an average importance to the target domain: $\kappa_i = \frac{1}{n_T} \sum_{j=1:n_T} k(x_{Si}, x_{Tj})$ and use a quadratic formulation found in [5] to evaluate the optimal coefficients β_i^*.

$$\beta^* \sim \arg\min_{\beta} \frac{1}{2}\beta^T K \beta - n_S \, \kappa^T \beta s.t. \; \beta_i \in [0, B] \; and \; \left| \sum_{i=1:n_S} \beta_i - n_S \right| \leq n_S \epsilon$$

where $K_{ij} = k(x_{Si}, x_{Sj})$. The second term in the optimization problem makes sure we choose β_i high when the average distance of X_{S_i} to all the X_{T_*} is low.

5 Experimental Settings and Results

Problem Settings: As per Figure 1 there are number of ways a prediction problem can be solved. For inductive transfer learning, we have two methods - *prior* and *multi-task*. In total the same prediction problem can be solved *via* 7 different methods. As a reminder, we denote the courses as follows: *6.002x* spring 2012 as A, *6.002x* fall 2012 as B, and *6.002x* spring 2013 as C. First we consider the *entire history* setting and then present results for *moving window* setting. For comparison we define an *a-posteriori* model.

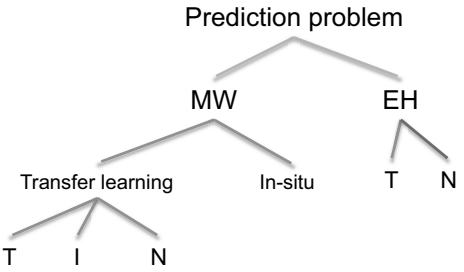

Fig. 1. Different ways we can learn and execute a model depending on the formulation we choose. MW stands for *Moving Window*, EH stands for *Entire History*, T stands for *Transductive model*, I stands for *Inductive model* and N stands for *Naive* model.

Definition 1. *a-posteriori model: It is the model built retrospectively using the labeled data from the target course itself. That is, the data from the target course is split into train and test and a model is trained on the training data and AUC is reported on the test data. We note that this is typically how results on dropout prediction are reported in several papers up until now.*

Entire History Formulation: Within the entire history setting, two different methods can be applied - *naive* and *importance sampling*. When applying these two methods we normalize features (using *min-max* normalization) independently within each course. Intuitively this makes sense, as normalizing all data using the normalization parameters calculated on the training set/previous course could induce misrepresentation of variables. For example, the variable *number of correct problems* could have different ranges as the total number of problems offered during a particular week may vary from offering to offering. Course-wise normalization allows us to compare learners to their peers thus making variables across courses comparable.

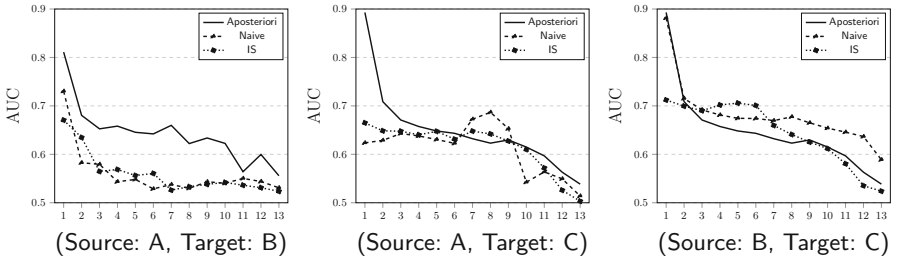

Fig. 2. Performance of transfer learning models for the *Entire History* formulation. Solid black line plots stands for the AUC obtained for the hypothetical case where we could access data from the future of the ongoing course (*a-posteriori* model). *Naive* method stands for the AUC obtained by naively using the source course as a training dataset and the target course as a test set. *IS* stands for *importance sampling* method.

Figure 2 shows the results achieved for three different transfer learning scenarios. On *x-axis* is the *lead* for the prediction problem and the *y-axis* is the average AUC across different amounts of history (same *lead* at different weeks). For example, at lead= 1, the *y-axis* value is the average of AUC values for 13 prediction problems defined at week 1-to-13. To summarize our results we use a question-answer format.

Is the Performance of an a-Posteriori Model a Good Indicator of Real Time Prediction Performance?: The a-posteriori model performance in most cases presents an optimistic estimate. Models trained on a previous offering struggle to achieve the same performance when transferred. This is especially true for one week ahead prediction. In two out of three cases in Figure 2 we notice a drop of 0.1 or more in AUC value when a model from previous offering is used. Hence, we argue that when developing *stopout* models for MOOCs for real time use, one *must* evaluate the performance of the model on successive offerings and report its performance.

Which is Better: Naive or Importance Sampling Based Transfer?: Based on our experiments we are not able to conclusively say whether *naive* transfer or the *importance sampling* based transfer is better. In some prediction problems one is better than the other and vice-versa. We hypothesize that the performance of *importance sampling* based approach can be further improved by tuning its parameters and perhaps fusing the predictions of both the methods can yield a better/robust performance.

What Could Explain the Slightly Better Performance of Transferred Models on C?: Number of learners in course C are significantly less than that in B or A. We posit that perhaps having more data in A or B enabled development of models that transferred well to C. This is specially true for B-to-C transfer. Additionally, since chronologically B is closer to C we hypothesize that not much may have changed in the platform or the course structure between these two offerings. This, together with the fact that B has more data enabled models trained on B to perform well on C (when compared to the *a-posteriori* model developed on C).

Moving Window Formulation. We next consider the *moving window* formulation. We consider that the target course is at week 4 and our goal is to predict whether or not learners who are in week 4 will stopout or will stay in the course. Fixing the `window size` to be 2 we can generate training examples for lead 1 by assembling covariates from week 1 and 2 and week 2 and 3. For lead 2 we can form training examples by assembling covariates from week 1 and 2. This allows us to generate labeled exemplars as we have knowledge about *stopouts* for week 3 and 4 in the ongoing course. Thus we are able to perform *in-situ* learning and inductive transfer learning methods. However, we are only able to solve two prediction problems - lead 1 and lead 2. We present the results achieved via multiple methods in Figure 3. We present the summary of our results.

How Did In-situ Learning Do?: In-situ learning did surprisingly well when compared to transfer learning methods under the *moving window* formulation.

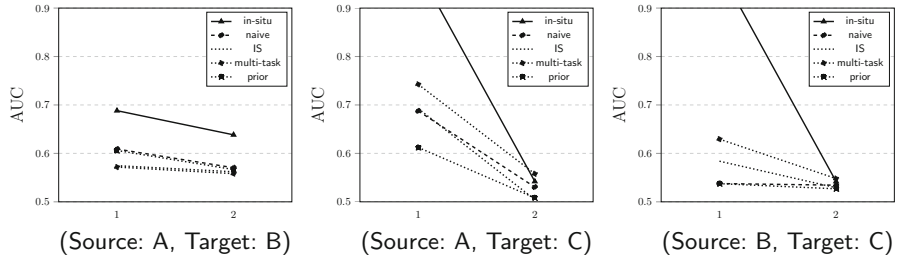

Fig. 3. Performance of transfer learning models for the *Moving Window* formulation

For both B and C the performance for lead 1 and 2 is between 0.6 -0.7 AUC (at week 4). This performance is also comparable to the *entire history* version of the same problem. In our subsequent work we intend to evaluate *in-situ* learning for a wide variety of prediction problems and not just at week 4.

6 Related Work

Dropout prediction and analysis of reasons for dropout are of great interest to the educational research community in a variety of contexts: e-learning, distance education, and online courses offered by community colleges. However, most studies focus on the identifying aspects of student's behavior, progress, and performance that correlate with dropout. These studies inform policy or give a broader understanding as to why students dropout [9,11]. Real-time predictions are rarely studied [6,7], but to the best of our knowledge, we have not found studies that examine whether models transfer well or if they could be deployed. Perhaps, MOOCs provide an ideal use case for transfer learning; with multiple offerings of the same course during a year, it is paramount that we use what we learned from the previous course to make predictions and design interventions in the next course. Within MOOC literature, there is an increasing amount of interest in predicting *stopout* [1,4]. In most cases, methods have focused on predicting one-week ahead and have not attempted to use trained on one course on another. Through this study, we are taking the first steps toward understanding different situations in which one can transfer models/data samples from one course to another. We have succeeded in (a) defining different ways real-time predictions can be achieved for a new offering, (b) multiple ways in which transfer learning can be achieved, and (c) demonstration of transfer learning performance.

7 Conclusion

In this paper we presented different methods that allow us to make real time predictions for learners in MOOCs. Particularly, we emphasized on transfer learning techniques that form the foundation for using models in real time. The key ideas of these methods were to formulate two different settings for a same prediction

problem (using aggregate data from previous course or partial data form the ongoing course) and implement advanced machine learning techniques for both.

From an engineering and scientific perspective, we believe these are first steps towards building high fidelity predictive systems for MOOC's (beyond retrospectively analyzing *stopout* prediction problem). To complete the model, we expect further work to designing new transfer learning methods, designing methodologies to tune the parameters for the current transfer learning approaches, evaluating the methods systematically on a number of courses, defining more covariates for learners and deploying the predictive system in an actual course.

References

1. Balakrishnan, G., Coetzee, D.: Predicting student retention in massive open online courses using hidden markov models. In: Technical report No. UCB/EECS-2013-109. EECS, University of California, Berkeley (2013)
2. Caruana, R.: Multitask Learning. Springer (1998)
3. Chelba, C., Acero, A.: Adaptation of maximum entropy capitalizer: Little data can help a lot. Computer Speech & Language **20**(4), 382–399 (2006)
4. Halawa, S., Greene, D., Mitchell, J.: Dropout prediction in moocs using learner activity features. In: Proceedings of the European MOOC Summit, EMOOCs (2014)
5. Huang, J., Gretton, A., Borgwardt, K.M., Schölkopf, B., Smola, A.J.: Correcting sample selection bias by unlabeled data. In: Advances in neural information processing systems, pp. 601–608 (2006)
6. Lauría, E.J.M., Baron, J.D., Devireddy, M., Sundararaju, V., Jayaprakash, S.M.: Mining academic data to improve college student retention: an open source perspective. In: Proceedings of the 2nd International Conference on Learning Analytics and Knowledge, pp. 139–142. ACM (2012)
7. Lykourentzou, I., Giannoukos, I., Nikolopoulos, V., Mpardis, G., Loumos, V.: Dropout prediction in e-learning courses through the combination of machine learning techniques. Computers & Education **53**(3), 950–965 (2009)
8. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Transactions on Knowledge and Data Engineering **22**(10), 1345–1359 (2010)
9. Street, H.D.: Factors influencing a learners' decision to drop-out or persist in higher education distance learning. Online Journal of Distance Learning Administration **13**(4) (2010)
10. Taylor, C., Veeramachaneni, K., O'Reilly, U.-M.: Likely to stop? predicting stopout in massive open online courses (2014). arXiv preprint arXiv:1408.3382
11. Tyler-Smith, K.: Early attrition among first time elearners: A review of factors that contribute to drop-out, withdrawal and non-completion rates of adult learners undertaking elearning programmes. Journal of Online learning and Teaching **2**(2), 73–85 (2006)
12. Veeramachaneni, K., O'Reilly, U.-M., Taylor, C.: Towards feature engineering at scale for data from massive open online courses (2014). arXiv preprint arXiv:1407.5238
13. Yang, Q., Pan, S.J.: Transfer learning and applications. In: Intelligent Information Processing, p. 2 (2012)