# Transfer Learning using Representation Learning in Massive Open Online Courses

Mucong Ding
Department of Computer Science and Engineering
Hong Kong University of Science and Technology
Hong Kong SAR, China
mcding@connect.ust.hk

Yanbang Wang
Department of Computer Science and Engineering
Hong Kong University of Science and Technology
Hong Kong SAR, China
ywangdr@connect.ust.hk

Erik Hemberg
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA, USA
hembergerik@csail.mit.edu

Una-May O'Reilly
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA, USA
unamay@csail.mit.edu

## ABSTRACT

In a Massive Open Online Course (MOOC), predictive models of student behavior can support multiple aspects of learning, including instructor feedback and timely intervention. Ongoing courses, when the student outcomes are yet unknown, must rely on models trained from the historical data of previously offered courses. It is possible to transfer models, but they often have poor prediction performance. One reason is features that inadequately represent predictive attributes common to both courses. We present an automated transductive transfer learning approach that addresses this issue. It relies on problem-agnostic, temporal organization of the MOOC clickstream data, where, for each student, for multiple courses, a set of specific MOOC event types is expressed for each time unit. It consists of two alternative transfer methods based on representation learning with auto-encoders: a passive approach using transductive principal component analysis and an active approach that uses a correlation alignment loss term. With these methods, we investigate the transferability of dropout prediction across similar and dissimilar MOOCs and compare with known methods. Results show improved model transferability and suggest that the methods are capable of automatically learning a feature representation that expresses common predictive characteristics of MOOCs.

## CCS CONCEPTS

• **Applied computing** → **E-learning**; • **Computing methodologies** → **Neural networks**; **Unsupervised learning**;


*LAK19, March 4–8, 2019, Tempe, AZ, USA*

ACM ISBN 978-1-4503-6256-6/19/03…$15.00
https://doi.org/10.1145/3303772.3303794

## KEYWORDS

Transfer Learning, Representation Learning, MOOC, Dropout Prediction, Dimensionality Reduction, Autoencoder

**ACM Reference Format:**
Mucong Ding, Yanbang Wang, Erik Hemberg, and Una-May O'Reilly. 2019. Transfer Learning using Representation Learning in Massive Open Online Courses. In *The 9th International Learning Analytics & Knowledge Conference (LAK19), March 4–8, 2019, Tempe, AZ, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3303772.3303794

## 1 INTRODUCTION

Massive open online courses (MOOCs) have become popular and provide an inexpensive, learner-directed learning environment. In MOOCs, predictive models of student behavior can support an instructor to improve the student learning, e.g., provide appropriate feedback and timely intervention. Multiple studies have proposed predictive models to, for example, analyze learning progress, support a better understanding of learning abilities, identify at-risk students, and indicate where pro-active interventions may be needed [7, 13].

Quantitative, behaviorally driven, predictive models have certain limitations. In particular, while they can be accurate, they may not be interpretable. For example, they do not integrate latent contextual information that is important for elaborating upon the observed behavior such as the fatigue or motivations of a student. Regarding dropout, they do not reveal the students who are not interested in attaining a certificate and others who may be autodidacts who ignore learning design patterns. Additionally, models do not integrate a teacher's goals and intended learning patterns which are complex factors that influence a learner. Nonetheless, while a course is ongoing and many students are enrolled while out of personal contact with the instructor, automated prediction can be helpful [2]. An accurate model can identify learners who could benefit from appropriate suggestive interventions which may prevent them from dropping out. It can assist in guiding learners with adaptive instructional materials and pathways to personally appropriate learning resources. Thus, herein, we focus on technical innovation that improves automation and predictive accuracy of models that are useful in transfer learning settings.

There are also technical and practical modeling challenges. Some models predict an outcome that occurs later in the course from the time a student's learning behavior is observed. One example is a model that predicts in week 3 whether a student will dropout in week 4. This type of model is trained, with machine learning, on data retrospectively collected from completed courses. This, however, makes it difficult to use in an ongoing course that differs from the completed course data. Different aspects between courses evolve, including the platform affordances and course design, while the learner cohort also differs. Despite best efforts, handcrafted features developed for the earlier offering may not express correlations in the ongoing course, i.e., they are brittle. Or, they may not exist in a subsequent offering, i.e., they are infeasible. For example, if a feature is defined on specific course exercises, it cannot be used in a subsequent offering if the exercises are removed. Previous work has relied on handcrafted features for transfer learning and had operational limitations regarding ongoing courses [4].

In this paper, our goal is to create operational predictive models for online use. We fundamentally ask whether it is possible, in general, to train a model on a source course's data and reliably transfer it to perform well for an ongoing course. One possible approach is to eliminate customized features engineered by a human that depend on domain knowledge and instead learn a latent representation amenable to the model transfer. Therefore, we propose to investigate transductive transfer learning methods [16], see step 2 and 3 in Figure 1. These methods assume that no label is available for the target task and instead learn a model from the target domain plus a source domain where there are labels and similarity to the target distribution. In a transductive approach known as representation-based transfer learning, a latent space representation is learned automatically using source and target data (the latter without labels).

Our particular research questions are:

(1) Does representation learning improve model transfer? We evaluate transferability within offerings for two courses and across two courses.

(2) Can representation-based learning work from a universal, basic set of MOOC activity features as input? We test a time-series per student where the frequencies of a set of specific MOOC activity types are expressed per time unit.

(3) Can transfer learning improve recognition of minority groups? If we group similar students and transfer learning for each group independently, does predictive performance improve?

(4) What are the embedded features that increase the transferability?

We employ a class of neural networks called auto-encoders (AEs) to compress the input into a latent space representation from which the input is reconstructed, as output. To perform well, the AE has to learn, through the training of its weights, to extract the most relevant features in the representation between the encoder and decoder. For transductive transfer when an AE is trained on both source and target features, its embedding is a set of lower dimensional features that compactly capture mutual properties between the source and target courses, which can then be used by subsequent modeling. For events with strong correlation, e.g. `play_video` and `pause_video` events, the auto-encoder (AE) learns a compressed

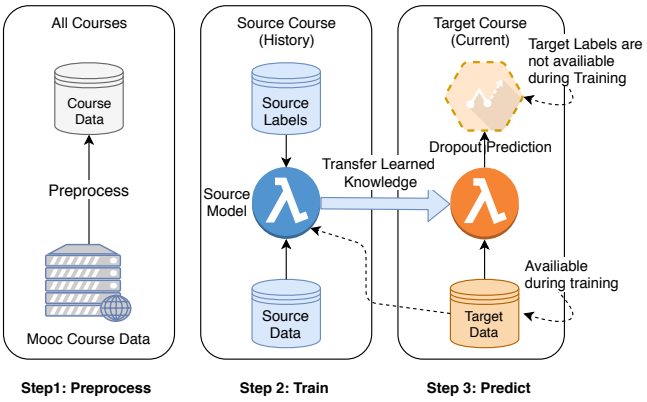



**Figure 1: Transfer learning work-flow.**

representation that reduces the noise and mutual-correlation between them. This leads to improved transferability. We investigate two variations of representation-based transfer learning: (1) **Post-processing the embedding by transductive principal component analysis**: a passive approach that learns the compact representation before it trains the predictive model. (2) **Training with correlation alignment loss**: an active approach that trains the auto-encoder and predictive model simultaneously.

With the learned representation of each method and the predictive model for the target, we can evaluate transferability and compare it with existing methods [4]. See Figure 1 for the work-flow of our approach. We choose for demonstration dropout prediction in six edX courses – three offerings of 6.00.1x "Introduction to Computer Science and Programming Using Python" and three offerings of 6.00.2x "Introduction to Computational Thinking and Data Science".

The paper is organized as follows. In Section 2, we present related work. In Section 3 we describe the courses, input data and dropout problem we use for demonstration. This sets up Section 4, where we present the two transfer learning methods. We evaluate the methods using the courses and dropout prediction problem in Section 5. In Section 6 we summarize and mention future work.

## 2 RELATED WORK

This section covers related work on transfer learning, representation learning with auto-encoders, feature learning and dropout prediction.

A summary and investigation into transductive transfer learning are provided in [1, 16]. There are two approaches for transductive feature learning: instance-based methods motivated by importance sampling, and representation-based methods using feature learning. In this paper, we explore representation based transfer methods and use the instance based algorithm as a baseline. A previous study of transfer learning for predictive models in MOOCs uses hand-crafted features and an importance sampling method to shift the source distribution towards the target one [4]. The study also proposes some inductive transfer methods by forcing models to learn from the features within a sliding window, but the resulting transfer methods use the previous week dropout as the target label

for the transfer setups and cannot predict the next week dropout well in an online course. A non-MOOC transfer learning study that investigates graduation rates in degree programs [11] uses *AdaBoost* and manual features.

Feature identification is a critical precursor to prediction [8]. Some human selected and engineered features are page views, video interactions, forum posts, and content interactions. By human and engineering we imply, respectively, that the choice of the combination is made by learning design experts or researchers and counts of the clickstream elements have to be combined to derive the feature. An extensive predictive modeling (and coincidentally dropout focused) investigation that relied upon feature engineering at scale on MOOC courses is [19]. The same study made use of crowdsourcing to engineer features. Herein we forgo feature engineering for feature learning, in the context of transfer learning. We are preceded by a study for MOOCs that used predictive engagement analytics with long short-term memory (LSTM) networks for feature learning [14].

Transfer learning is applicable to predicting any outcome variable, and we demonstrate it with dropout prediction. There are many types of predictive models for MOOCs, e.g., dropout, certification and grade [8, 9]. There are a variety of related papers which investigate the dropout prediction problem [5–7, 13]. Most existing work use machine learning approaches such as logistic regression (LR), support vector machine (SVM), decision trees (DT) and Neural Networks (NN).

## 3 PROBLEM DESCRIPTION

In this section, we compare and contrast the courses we use to evaluate the representation-based transfer learning methods we purposed. We describe how we organize student activity data collected from them for input to the methods. Finally, we define the predictive modeling problem, dropout, that we use to demonstrate the transfer learning methods on.

### 3.1 Input Data Organization

In practical online prediction, some student attributes are unavailable (e.g., certificate and registration information) and cannot be used to filter the set of students. Thus, the data used for prediction exhibits high variance. We use the click-stream log events, which were engineered by the learning platform developers without a prediction problem in mind, as input to the transfer learning methods. The list of events can be found in the legend of Figure 6. All events have a *time-stamp* and *event type*. Therefore it is straightforward to aggregate the events by week and event type, per student. This results in a multivariate time series, for each student, $[\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_T]$, where each $\mathbf{x}_k$ is a vector of the normalized frequencies of the event types in that week $k$, and $T = 9$ is the number of weeks. The sets of event types of courses do not need to be identical. However, the most frequent and important event types often overlap.

### 3.2 Courses

We experiment with two courses offered on the edX MOOC platform: *Introduction to Computer Science and Programming Using Python* (6.00.1x), and *Introduction to Computational Thinking and Data Science* (6.00.2x). We have 3 offerings of each course.

| Course | Assignment | Due Week |
|---|---|---|
| **6.00.1x** | Python Basics | Week 4 |
| | Simple Programs | Week 5 |
| | Structured Types | Week 7 |
| | Good Programming Practices | Week 8 |
| | Object Oriented Programming | Week 9 |
| **6.00.2x** | Optimization | Week 5 |
| | Randomness | Week 6 |
| | Midterm Exam | Week 7 |
| | Statistics | Week 8 |
| | Modeling and Fit | Week 9 |

**Table 1: 6.00.1x and 6.00.2x Assignments.**

| Course | N. Videos | N. Finger Exercises |
|---|---|---|
| **6.00.1x** | 81 | 555 |
| **6.00.2x** | 43 | 177 |

**Table 2: Resource quantities in terms of video and finger exercises. 6.00.1x has more resources than 6.00.2x.**

In terms of structure, all offerings have 9 weeks. Both courses have multiple units, where each unit has an associated graded problem set. Students are expected to watch lecture videos narrated by instructors and complete "finger exercises" - optional problems interspersed in lecture videos that teach the content discussed in the video. The topics of each course differ because one course is the continuation of the other, see Table 1 for details. The quantities of videos and finger exercises are much higher in 6.00.1x, see Table 2. Enrollment and activity volume measured in click-stream events are shown in Table 3.

We can statistically compare the courses to gain a mathematical estimation of similarity. We use the Proxy A-distance (an approximation of the $\mathcal{H}$-divergence [3]) as an indicator of the similarity between two samples where each sample distribution consists of the frequency of events with an arbitrary type that occurred in an arbitrary week of a specific course. The pair-wise Proxy A-distances (PADs) are difficult to visualize, so we first compute a 2D-embedding where the distances between different distributions of different courses are the average PADs. We find the 2D-embedding which best preserves the distances by multi-dimensional scaling. We visualize the pairwise distances in Figure 2. This identifies 6.00.2x Spring 2016 (*2A*) and 6.00.2x Fall 2016 (*2B*) as most similar while 6.00.1x Summer 2016A (*1A*) and 6.00.2x Fall 2016 (*2B*) are most dissimilar. This is mainly because *2A* and *2B* are two offerings of the same course, while *1A* is a different course.

### 3.3 Dropout Prediction

We adopt a widely used dropout definition: dropout occurs when the student no longer interacts with the MOOC platform. When considering all click-stream event types, the dropout labels become noisy. Therefore we define dropout based on video events (e.g., `play_video`). For a time-granularity based on weeks, the dropout week of a student is defined as the week after the student's last video interaction event. By this definition, students cannot drop

| Course | 6.00.1x | | | 6.00.2x | | |
|---|---|---|---|---|---|---|
| Offering | Summer 2016A(*1A*) | Summer 2016B(*1B*) | Spring 2017(*1C*) | Spring 2016(*2A*) | Fall 2016(*2B*) | Spring 2017(*2C*) |
| N. students | 37,363 | 15,199 | 26,011 | 6,774 | 6,945 | 5,893 |
| N. events | 17,333,974 | 7,900,908 | 13,176,220 | 2,642,528 | 2,501,276 | 2,034,539 |

**Table 3: Summary of the course statistics regarding number of students and events for 6.00.1x and 6.00.2x. We use the symbols in brackets to note the six courses through the rest of the paper.**

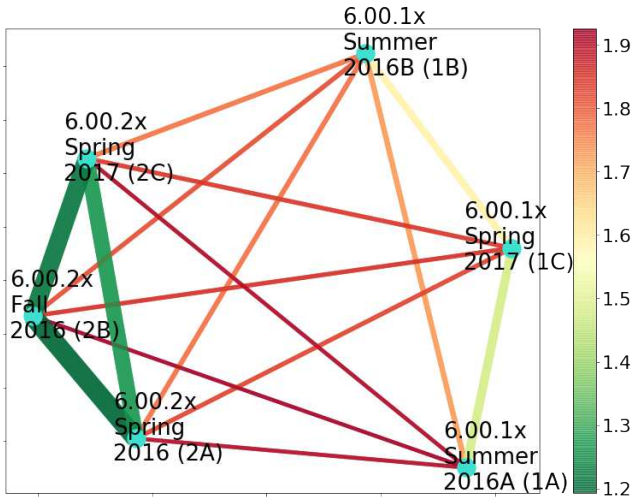



**Figure 2: 2D-embedding by MDS of the Proxy A-distances (PADs) between features from different courses. The color-bar shows the distance value. The color and thickness of a line segment are proportional to the pairwise distance (Note, the range of PAD is defined in $[0, 2]$).**

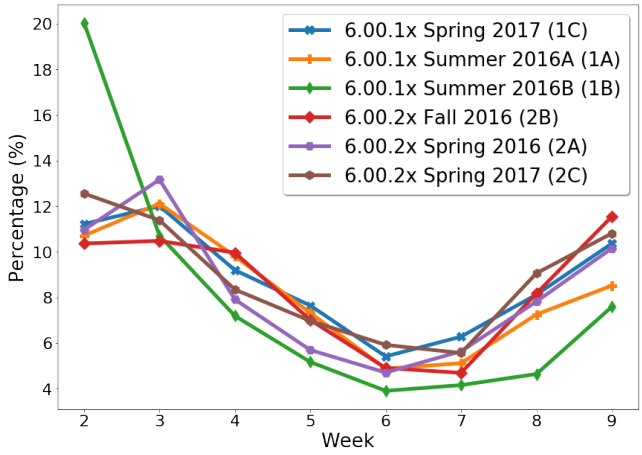



**Figure 3: Percentage of students dropout in each week. X-axis shows the week and Y-axis the percentage of students that dropped out, the color shows the course.**

out in the first week, and we train one predictive model for each week after the first. The percentages of student dropout in each week are shown in Figure 3.

Prediction can be based on data from the first week up to the week before the prediction. A student is characterized by a pair of time-series $(\mathbf{x}, \mathbf{y})$. The label of a student is a uni-variate time-series $\mathbf{y} = [y_1, \cdots, y_T]$, where $y_k \in \{\text{False}, \text{True}\}$ indicates whether the student has dropped-out in week $k$ and $T$ is the total number of weeks. The features of a student, can be represented as a multi-variate time series $\mathbf{x} = [\mathbf{x}_1, \cdots, \mathbf{x}_T]$, where $x_k$ is a set of features in week $k$. To predict student dropout during the course, we train $T-1$ models $[f_2(\cdot), \cdots, f_T(\cdot)]$, one for each week, where $f_k(x_k)$ is the prediction for week $k$. The model $f_k(\cdot)$ for week $k$ uses $y_k$ as label, and $[\mathbf{x}_1, \cdots, \mathbf{x}_{k-1}]$ as features. Each model solves a binary classification problem, i.e. did the student stay in the course or dropout.

## 4 TRANSFER LEARNING

In this section, we formulate the problem of transferring models between MOOC courses and then introduce our transfer methods.

### 4.1 Transfer Learning Definition

A domain $\mathcal{D}$ consists of two components: a feature space $\mathcal{X}$ and a marginal probability distribution $\Pr(X)$ where $X = \{x_i\}_{i=1}^n \subset \mathcal{X}$. Given a specific domain, a task $\mathcal{T}$ consists of two components: a label space $\mathcal{Y}$ and a conditional probability distribution $\Pr(Y|X)$ where $Y = \{y_i\}_{i=1}^n \subset \mathcal{Y}$. Considering a source course $S$ and target course $T$, the feature spaces of the source and target domains $\mathcal{D}_S$ and $\mathcal{D}_T$ are the same but the feature distributions are different $\Pr(X_S) \neq \Pr(X_T)$. However, the prediction tasks $\mathcal{T}_S$ and $\mathcal{T}_T$ for the two domains are the same, as the conditional distributions coincide $\Pr(X_S|Y_S) = \Pr(X_T|Y_T)$.

**Definition 1. Transductive Transfer Learning Problem in MOOCs** In transfer learning the training of the target predictive function $f_T(\cdot)$ in the target domain $\mathcal{D}_T$ is supplemented using the knowledge in the source domain $\mathcal{D}_S$, where $\mathcal{D}_S \neq \mathcal{D}_T$ and $\mathcal{T}_S = \mathcal{T}_T$. At training time, source data $\{x_{S_i}, y_{S_i}\}_{i=1}^{n_S}$ and the unlabeled target domain features $X_T = \{x_{T_i}\}_{i=1}^{n_T}$ are available.

### 4.2 Transfer Learning Methods

We use auto-encoders to learn a representation space that is common to both source and target domains. By training an auto-encoder (AE) on both of the source and target features (and, in some variants, the source labels) and using a learning signal that measures the output's distance from the input, the AE's embedding layer between the encoder and decoder is forced to capture the common characteristics of the two distributions. There is a trade-off between

the prediction model's capability and the dimensionality of the representation space. When the reduction is small, the auto-encoder can learn multiple modes in the distribution but the predictor is prone to overfit to the source task. When the reduction is large, the auto-encoder can only learn a single mode presenting the risk that the embedding is not predictive enough. Thus a dimensionality should be carefully chosen for a specific combination of model and data set.

We introduce two different transfer learning methods: *1)* Passive Transfer with Transductive-PCA (**Passive-AE Transfer**) *2)* Active Transfer with CORAL loss (**Active-AE Transfer**)

*4.2.1 Passive Transfer with Transductive PCA (T-PCA).* The workflow is depicted in Figure 4. The target embedding is obtained in a "passive" way, since no objective functions are defined and used. First, an AE is trained on both the source and target features. Next, (Step 2 in Figure 4), using target features only, a PCA [12] transform is fit on the learned target embedding to transform the source embedding for predictive model training (Step 3) [15]. This avoids the prediction model being trained on an embedding that has learned irrelevant features from the source domain. The number of outputs of T-PCA is set to be larger than the number of predicted labels.

*4.2.2 Active Transfer with CORAL loss.* An unsupervised transfer learning method, CORAL performs a transformation to align the second-order statistics of the source and target domains [17]. This removes variations that are only present in the source domain in the learned embedding. A general transform is achieved a by deep neural network [18]. A CORAL loss is introduced as a term in the objective function because minimizing the prediction loss itself can lead to overfitting to the source domain, causing reduced performance on the target domain. Let $C_S$ and $C_T$ indicate the covariance matrices of the source and target embedding $E_S = \{\mathbf{u}_i\}_{i=1}^{n_S}$ and $E_T = \{\mathbf{v}_i\}_{i=1}^{n_T}$, the CORAL loss [18] is defined as $\mathcal{L}_{CORAL} = \frac{1}{4d^2}\|C_S - C_T\|_F^2$ where $\|\cdot\|_F$ denotes the Frobenius norm, and $d$ is the dimension of the common embedding space $\mathbf{u}, \mathbf{v} \in \mathcal{E} \subseteq \mathbb{R}^d$. While minimizing the CORAL loss alone can lead to degenerated features, jointly training the auto-encoder and the embedding predictor with both losses and also the target reconstruction loss tries to strike a balance (Figure 5).

## 4.3 Auto-Encoders and Predictive Models

Both transfer learning methods use an AE and a predictive model, and for both autoencoding and prediction, neural networks can be used. In preliminary AE studies, we used a neural network implementation of PCA as an AE baseline. We then explored deep networks: a multi-layer perceptron, a convolutional neural network (CNN) (the order of course materials in each week preserves the implicit dependence in the knowledge graph, so we can apply 1D-convolution in the time domain to find the implicit learning patterns) and a Long Short-Term Memory (LSTM) network adapted for autoencoding. The LSTM AE has two 1D-convolutional layers with kernel size 1 and a bi-directional LSTM layer in the encoder and one 1D-convolutional layer with kernel size 1 and a bi-directional LSTM layer in the decoder (see Appendix A for details). In a bi-directional LSTM, two hidden layers of opposite directions are connected to the same output to make future input information reachable from the current state. We report our best results which use the LSTM AE.

Training models for prediction relies on the embedded features from the AE (or T-PCA) representations. We explored predictive modeling both with and without embedding features. We used logistic regression (LR) as a baseline of the predictive models and then compared using CNN and LSTM with architectures similar to the encoder parts of AEs. We report our best results which use CNN on top of embeddings and LSTM without embeddings (see Appendix A for details).

# 5 EXPERIMENTS

We performed a massive number of experiments, which was made possible by using an efficient and scalable software framework that allows model tuning. Preliminary experiments allowed us to settle on parameters, e.g., architectures of the networks and training hyper-parameters to use throughout evaluations. We ran our entire pipeline multiple times with different combinations of models and architectures. The pipeline is released in an open-source `Python` MOOC learner data science analysis (`mldsa`) toolkit [1].

We use the area under the receiver operating characteristic curve (AUC) to measure the performance of all predictive models. The AUC scores are chosen from the best performing architecture and are averaged on all possible pairs of source and target courses and on all weeks. See Appendix B for detailed experimental configurations.

Our input features are based on the thirteen common event types, see Figure 6. These events all tend to decrease in frequency, most likely due to dropout. We also note that some events are highly correlated (e.g. `problem_check` and `problem_graded`). This is one of the motivations of representation learning on the raw features.

## 5.1 Transfer Learning Baselines

We set up four baseline models. All of them use LSTM or LSTM AE and CNN for prediction (see Appendix A), the same as **Passive-AE Transfer** and **Active-AE Transfer**.

**Label-Truth and Label-Truth-AE** These models do not use transfer learning. Instead, they are trained with target features and labels (retrieved retrospectively) and provide a label truth baseline. **Label-Truth-AE** uses an LSTM AE embedding for prediction. The target domain data is split into train and test, and test accuracy is reported.

**Naive Transfer** This model is learned on the source course (without using target labels) and subsequently applied to the target course.

**In-Situ Learning** We learn a predictive model from the data from the on-going course itself with a sliding window [4]. For week $k \leq 3$, we use window size $w = k - 2$ and apply the model trained for the previous week $k - 1$ for prediction. It transfers between weeks of the same target course and does not use the source course.

**Instance-Based Transfer** Here it is assumed the features of the source and target domains are drawn from a common distribution and the difference comes from a sample selection bias. We correct this bias by giving more weight to the students in the source course that are similar to the students in the target course.

[1] https://github.com/MOOC-Learner-Project/MOOC-Learner-Data-Science-Analytics

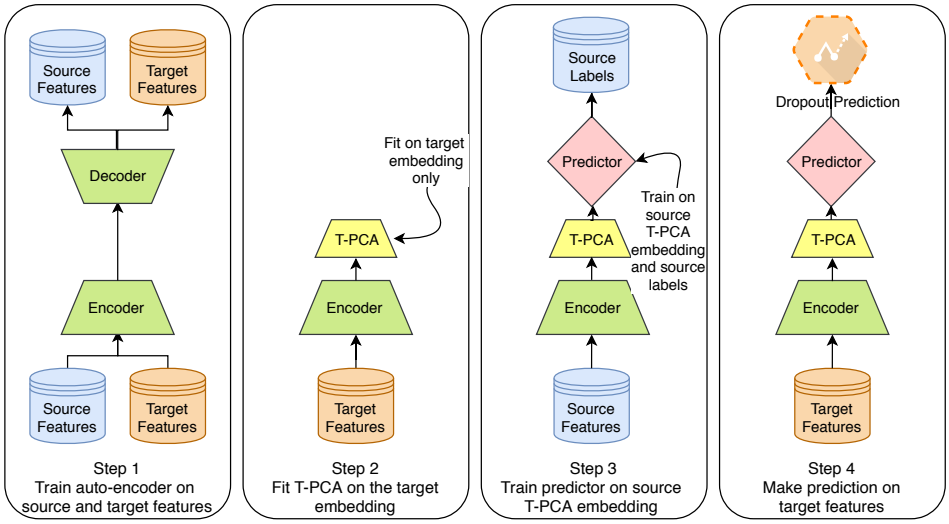



**Figure 4: Passive transfer with transductive PCA.**

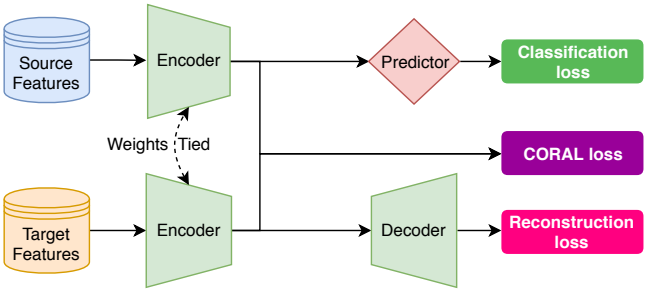



**Figure 5: Active transfer with CORAL loss.**

The learning objective in the target domain $\Sigma_{i=1}^{n_T} l(x_{T_i}, y_{T_i}, \theta)$ is approximated by $\Sigma_{i=1}^{n_S} \Pr(x_{T_i})/\Pr(x_{S_i}) l(x_{S_i}, y_{S_i}, \theta)$. Thus, by applying different loss weights $\Pr(x_{T_i})/\Pr(x_{S_i})$ to each instance $x_{S_i}$ in the source domain we can train a precise model for the target domain. We use the kernel-mean matching algorithm [10] to compute the weights.

## 5.2 Model Selection

First, without transfer, we compared the prediction performance of the LSTM neural network with the standard logistic regression (LR), see Figure 7. The LSTM model consistently outperforms LR by a large margin. Thus we chose LSTMs.

Next, we compared the performance of predictive models using the embeddings from the AEs as input. More specifically, a CNN model trained on an embedding learned by an LSTM AE or PCA, see Figure 7. With bottleneck size of eight, the AUC scores of the CNN and LSTM AE embedding are similar to the LSTM's. This implies that it is possible to learn a compact and effective embedding, and suggests the effective use of dimensionality reduction for transfer learning. All predictive models perform better in the middle part of the course, and their AUCs are negatively correlated with the dropout rates (see Figure 3).

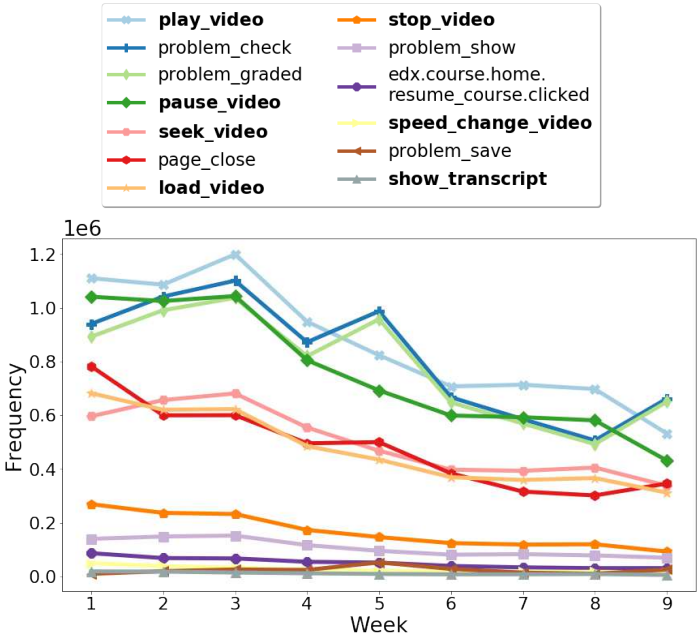



**Figure 6: The total frequencies of the thirteen common event types used to define the raw features in all the six courses. X-axis shows the week and Y-axis the frequency, the color shows the event type. Bold indicates video event. The events tend to decrease and some are correlated.**

## 5.3 Transfer Learning Results

We now compare the proposed transfer learning methods with the baselines and **Label-Truth** case. The transfer methods that perform best are the **Passive-AE Transfer** and **Active-AE Transfer**, even similar to the **Label-Truth** baseline. The dropout prediction

| | *1B→1A* | *1C→1A* | *1A→1B* | *1C→1B* | *1A→1C* | *1B→1C* | Avg. |
|---|---|---|---|---|---|---|---|
| **Passive-AE Transfer** | .758±1 | .777±1 | .749±2 | .757±2 | .801±1 | .784±2 | .771±2 |
| **Active-AE Transfer** | **.759±1** | **.788±1** | **.769±2** | **.769±2** | **.812±1** | **.792±2** | **.782±1** |
| **In-Situ Learning** | .681±2 | | .659±2 | | .724±2 | | .688±2 |
| **Instance-Based Transfer** | .613±1 | .683±1 | .608±1 | .663±1 | .653±1 | .649±1 | .645±1 |
| **Naive Transfer** | .716±1 | .756±1 | .700±3 | .699±3 | .743±2 | .736±2 | .725±2 |
| **Label-Truth** | .800±1 | | .773±1 | | .819±1 | | .797±1 |

**Table 4: 6.00.1x→6.00.1x: Average AUC scores of transfer methods and the Label-Truth (no-transfer) baseline for all weeks. Transfer between offerings of the same course.**

| | *2A→1A* | *2B→1A* | *2C→1A* | *2A→1B* | *2B→1B* | *2C→1B* | *2A→1C* | *2B→1C* | *2C→1C* | Avg. |
|---|---|---|---|---|---|---|---|---|---|---|
| **Passive-AE Transfer** | **.753±1** | **.752±1** | **.760±1** | **.735±2** | .733±2 | **.745±2** | **.786±1** | **.777±1** | **.782±1** | **.758±1** |
| **Active-AE Transfer** | .713±1 | .720±1 | .717±1 | .734±2 | **.740±2** | .743±2 | .755±1 | .751±2 | .757±2 | .737±2 |
| **In-Situ Learning** | .681±2 | | | .659±2 | | | .724±2 | | | .688±1 |
| **Instance-Based Transfer** | .590±2 | .569±2 | .606±1 | .637±3 | .640±2 | .600±1 | .579±2 | .528±1 | .598±3 | .594±2 |
| **Naive Transfer** | .668±1 | .700±2 | .706±2 | .675±3 | .684±3 | .676±3 | .735±2 | .716±2 | .739±2 | .702±2 |
| **Label-Truth** | .800±1 | | | .773±1 | | | .819±1 | | | .797±1 |

**Table 5: 6.00.2x→6.00.1x: Average AUC scores of transfer methods and the Label-Truth (no-transfer) baseline on all weeks. Transfer between courses.**

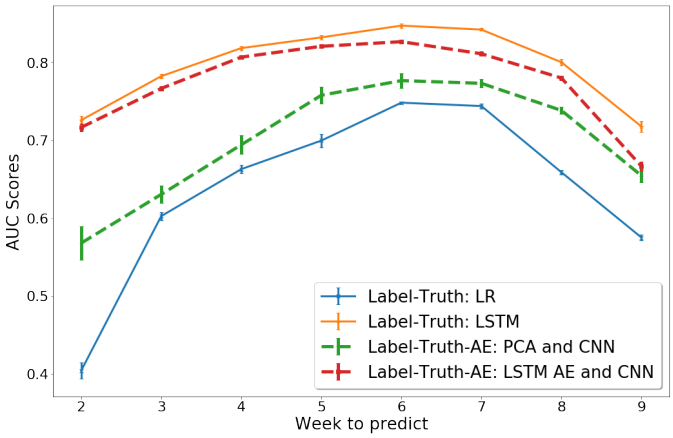
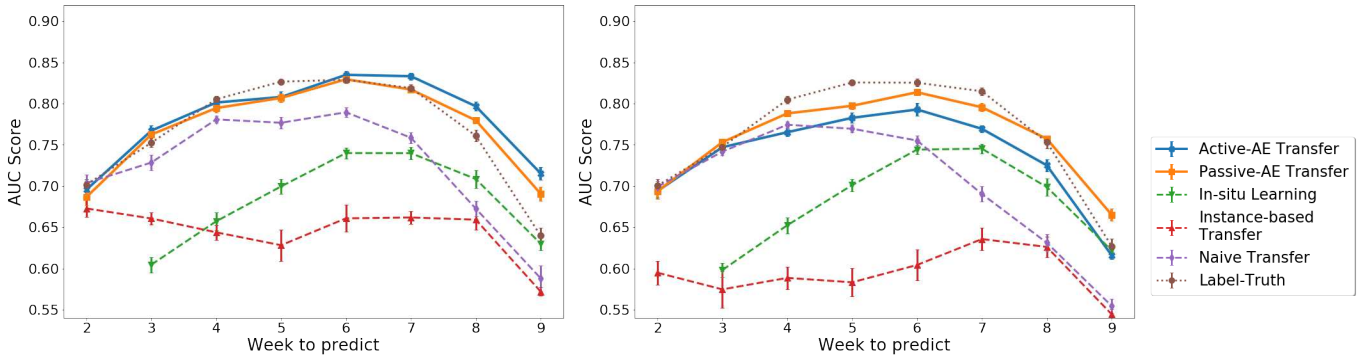



**Figure 7: Average AUC scores (with error bars) of predictors of different architectures on the six courses over different weeks. We can see LSTM consistently outperforms LR, and the prediction performance of CNN on embedding learned by LSTM AE (which consists of eight features per week) is close to the best performance of predictors on raw features.**

performance of **Passive-AE Transfer** and **Active-AE Transfer** transfer with the baselines (**Label-Truth**, **Label-Truth-AE**, **Naive Transfer**, **In-Situ Learning**, and **Instance-Based Transfer**), for the similar pairs of source and target 6.00.1x→6.00.1x (within offerings of one course) are shown in Table 4 and Figure 8a, and the dissimilar pairs 6.00.2x → 6.00.1x (across two courses) in Table 5 and Figure 8b. Note that the performance of **In-Situ Learning** and **Label-Truth** does not depends on the choice of source course. All transfer baselines have significantly lower average AUCs than the **Passive-AE Transfer** and **Active-AE Transfer** methods.

Now we analyze the average AUC per predicted week, see Figure 8. It shows that **Naive Transfer** overfits to the source domain from week 5. **In-Situ Learning** learning does not work well until the final weeks of the course since it applies a sliding window and a considerable proportion of information is lost when week $k$ is small. **Instance-Based Transfer** has abysmal performance on all weeks, since the raw features we used are high-dimensional and sparsely distributed, and hence obtaining a good approximation of the sampling weights is very challenging. We note that **In-Situ Learning** and **Instance-Based Transfer** struggles to outperform **Naive Transfer** in most cases. An important reason for this is that we used a very low-level input representation (time-series of click-stream events). These methods need complex features whereas representation learning is capable of learning from simpler ones. We anticipate an improved performance for the baseline transfer methods with handcrafted features, but that is not certain and also demands a human to solve the challenge of engineering good features.

The sensitivity of some of the important parameters of the **Passive-AE Transfer** and **Active-AE Transfer** methods are also investigated in Figure 9. We analyze how the performance of passive and active transfer is correlated to the PAD and the sample size ratio between the source and target courses. We see that the **Passive-AE Transfer** performs better than the **Active-AE Transfer** approach when transferring from a course with $\sim 6,000$ students (6.00.2x) to one with $\sim 15,000 - 40,000$ students (6.00.1x) (the upper right corner of Figure 9), and in the other cases the **Active-AE Transfer** often sightly outperforms the **Passive-AE Transfer**. We find that the best transfer performance is close to the **Label-Truth** baseline when both the target to source sample size ratio and the PAD




**(a) Average AUCs on the six combinations of similar courses (6.00.1x→6.00.1x).**

**(b) Average AUCs on the nine combinations of dissimilar courses (6.00.2x→6.00.1x).**

**Figure 8: Average AUC scores (with error bars) of transfer methods and the Label-Truth (no-transfer) baseline for each week on different groups of source and target combinations.**

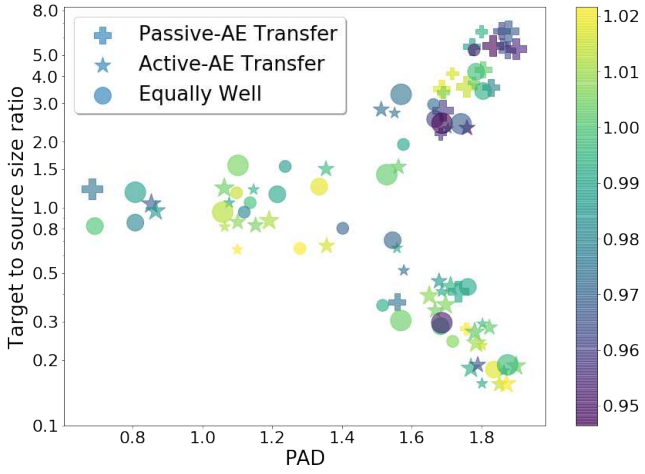



**Figure 9: Scatter plot of all Passive-AE Transfer and Active-AE Transfer experiments. X-axis shows the Proxy A-distances (PAD) between the source and target input features (sub-sequences till week $k-1$ when predicting for week $k$). Y-axis shows the ratios between the number of students till week $k-1$ in the target and source course. The shape shows the wining method among Passive-AE Transfer and Active-AE Transfer. If the difference between their AUCs is less than 1% of their averages, we consider they perform equally well. The color shows the ratio between its AUC and Label-Truth's. The size of a point positively correlates with the length of input feature sequence (prediction week $k$).**

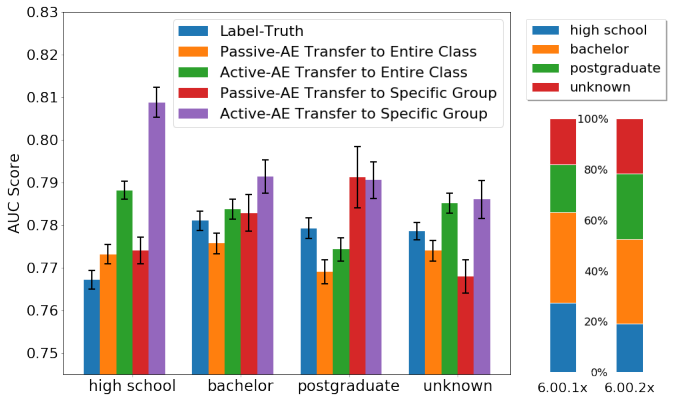
are small. **Active-AE Transfer** or **Passive-AE Transfer** transfer have on average an 8% improvement compared with the **Naive Transfer** baseline in terms of AUC scores (calculated from the last columns of Table 4 and Table 5).

## 5.4 Experience-Based Transfer Models

The composition of a MOOC class can be very diverse, and this is one of the differences compared to traditional education. Since 6.00.2x is a more in-depth course than 6.00.1x, the proportion of high school students in 6.00.2x is smaller, but there are more post-graduate students (Figure 10b). Training a predictive model for high school students on 6.00.2x can be difficult in the sense that there are not enough samples to learn from. Transferring the knowledge from 6.00.1x by the proposed methods can partially solve this problem. Equipped with the student background labels, there are two possible ways of transfer learning: transfer to the entire target class as we did in the other experiments, or consider each group of students as a target and transfer specifically to that group in an experiment. With simpler target distributions and thus easier transfer objectives, the latter approach is potent to achieve better prediction performance. We evaluate their performance when transferring from 6.00.1x to 6.00.2x as shown in Figure 10a. Where we see the average AUC on high school students is improved by active transfer, and is even further improved if we specifically transfer to the high school group. **Passive-AE Transfer** does not perform as well as **Active-AE Transfer**, since the source course has much more students than the target. Results showed that **Active-AE Transfer** to specific groups performed the best. This approach can also be applied to minority groups based on other demographic variables including income level, gender, age, location, ethnicity, and race.

## 5.5 Examining the Embeddings

To further examine the transfer embeddings we replaced the **LSTM-AE** used by **Active-AE Transfer** with a neural network implementation of PCA to access the embedding as a linear transformation of the raw feature space. Now we can calculate the "weight" (the scaling factor in the direction of a feature under the PCA transformation) of a raw feature in the learned embedding, which is an indicator of how vital the raw feature is for the transfer learning task. We calculate the average weights of the thirteen features on different groups of source and targets, as shown in Figure 11. The average weights on all possible transfers (the last bar) show the




**(a) Average AUCs on students with different educational backgrounds in 6.00.2x.**

**(b) Class compositions.**

**Figure 10: (a): Averages AUC scores on students with different educational backgrounds for the Label-Truth case, and the Passive-AE Transfer and Active-AE Transfer methods when transferring from 6.00.1x to an entire 6.00.2x offering or a specific group of students within that offering on all weeks on all possible source and target combinations. (b): Average percentages of students with high school, bachelor, and postgraduate degrees in 6.00.1x and 6.00.2x.**

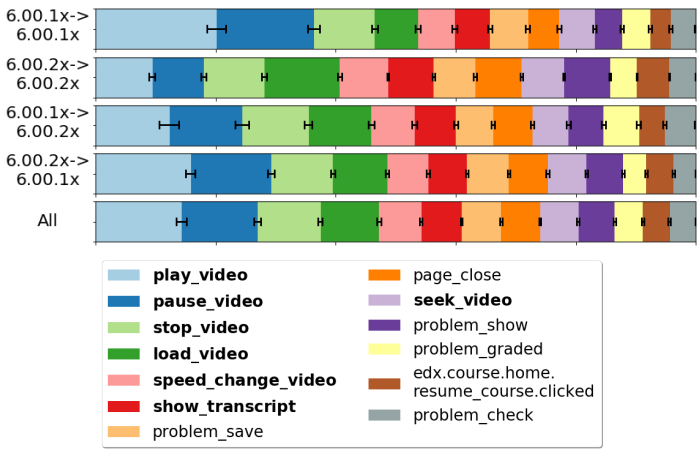



**Figure 11: Compositions of the embeddings learned by Active-AE Transfer (with error bars) using the neural network implementation of PCA. The legends are ordered with decreasing importance in the average embedding (the last bar).**

relative importance of each raw features, and we order the legend labels accordingly. We can see that video related events (bold) occupy the first six places, which implies they are more transferable and predictive than the others.

## 6 CONCLUSIONS

In this paper, we explored the possibilities of using deep autoencoders for transfer learning among courses in MOOCs. We proposed two methods to improve the transferability of learned embedding: the **Passive-AE Transfer** using transductive PCA to remove the variations present in the source domain but irrelevant for the target domain, and the **Active-AE Transfer** using the CORAL loss to force the alignment of the second-order statistics of the source and target embeddings. Moreover, the deep transfer models solve the domain-specific feature engineering problem and can learn compact and effective representations from the raw features, a straight-forward representation of the click-stream. We found that our transferred models consistently outperform the other transfer baselines and achieve similar performance compared with the label-truth (no transfer) models which are trained on the target course. In this sense, we believe that we have made significant progress in solving the transfer learning problem in MOOCs. With the acknowledged limitations that the models, while accurate and automated, are not transparent and do not integrate contextual human knowledge like the learning design.

We answered the following research questions: (1) Does representation learning improve model transfer? We evaluated transferability within offerings of two courses and across two courses. We found that transferring from a course with fewer students to one with more students **Passive-AE Transfer** was best versus **Active-AE Transfer** and baselines. In the opposite situation, **Active-AE Transfer** was best versus **Passive-AE Transfer** and baselines. Both passive and active methods approached the AUC level of the **Label-Truth** (no transfer) baseline, which learned from target labels directly. (2) Can representation-based learning work from a universal, basic set of MOOC activity features as input? We successfully used the same time series per student where the frequencies of a set of specific MOOC activity types are expressed per time unit as input to every experiment. It supported our transfer learning results. This suggests that, for transfer learning problems in MOOCs, in general, it is possible to eliminate costly feature engineering. (3) Can transfer learning improve recognition of minority groups? If we group similar students and transfer learning for each group independently, does predictive performance improve? We grouped students by their highest level of education and found that transfer learning with the proposed methods can help improve the prediction performance on minority groups, and transferring specifically to a target group might achieve even higher performance. This is an example where contextual knowledge, if available, can be used to improve this methodology despite that knowledge not being explicitly integrated into the model. (4) What are the embedded features that increase the transferability? We calculated the weights of event features in the embedding learned by **Active-AE Transfer** using PCA instead of LSTM AE and showed that video related events are more transferable and predictive than the others.

The contributions of this paper are:

(1) We introduced two online transfer learning methods based on representation learning that improve prediction for the target course and eliminate manual feature engineering. We improve the dropout prediction AUC scores by 8% using either method compared with the naive transfer baseline.

(2) We introduced a data organization for input to the prediction and transfer methods that requires no feature extraction. It is a time series per student where the frequencies of a set of specific MOOC activity types are expressed per time unit.

(3) Through visualization and metric analysis, we described the representation-based learning embeddings.

(4) We found that transfer learning for specific groups of students independently improved predictions, facilitating more specific learning support.

For future work, a direct extension of our work is to apply the transfer algorithms to different types of MOOCs and across MOOC platforms. It will also be interesting to investigate how we can effectively learn from multiple source courses, simultaneously characterizing and utilizing the sources' relationships to the target course based on course content and student population. Finally, it is possible to investigate further raw data representations that take structural course context into account, not only temporal activity.

## REFERENCES


[1] Andrew Arnold, Ramesh Nallapati, and William W Cohen. 2007. A Comparative Study of Methods for Transductive Transfer Learning. In *ICDM Workshops*. ICDM, Pittsburgh, PA, USA, 77–82.

[2] Ryan Shaun Baker and Paul Salvador Inventado. 2014. Educational data mining and learning analytics. In *Learning analytics*. Springer, New York, NY, USA, 61–75.

[3] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Machine learning* 79, 1-2 (2010), 151–175.

[4] Sebastien Boyer and Kalyan Veeramachaneni. 2015. Transfer Learning for Predictive Models in Massive Open Online Courses. In *Artificial Intelligence in Education*, Cristina Conati, Neil Heffernan, Antonija Mitrovic, and M. Felisa Verdejo (Eds.). Springer International Publishing, Cambridge, MA, USA, 54–63.

[5] Christopher Brooks, Craig Thompson, and Stephanie Teasley. 2015. A time series interaction analysis method for building predictive models of learners using log data. In *Proceedings of the fifth international conference on learning analytics and knowledge*. ACM, 126–135.

[6] Concepción Burgos, María L. Campanario, David de la Peña, Juan A. Lara, David Lizcano, and María A. Martínez. 2018. Data mining for modeling students' performance: A tutoring action plan to prevent academic dropout. *Computers & Electrical Engineering* 66 (2018), 541 – 556.

[7] Devendra Singh Chaplot, Eunhee Rhim, and Jihie Kim. 2015. Predicting Student Attrition in MOOCs using Sentiment Analysis and Neural Networks.. In *AIED Workshops (CEUR Workshop Proceedings)*, Jesus Boticario and Kasia Muldner (Eds.), Vol. 1432. CEUR-WS.org, Seoul, South Korea.

[8] Josh Gardner and Christopher Brooks. 2018. Student success prediction in MOOCs. *User Modeling and User-Adapted Interaction* 28, 2 (2018), 127–203.

[9] Josh Gardner, Christopher Brooks, Juan Miguel Andres, and Ryan Baker. 2018. Replicating MOOC Predictive Models at Scale. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale (L@S '18)*. ACM, New York, NY, USA, 1:1–1:10. https://doi.org/10.1145/3231644.3231656

[10] Jiayuan Huang, Alexander J. Smola, Arthur Gretton, Karsten M. Borgwardt, and Bernhard Scholkopf. 2006. Correcting Sample Selection Bias by Unlabeled Data. In *Proceedings of the 19th International Conference on Neural Information Processing Systems (NIPS'06)*. MIT Press, Cambridge, MA, USA, 601–608.

[11] Xin J. Hunt, Ilknur Kaynar Kabul, and Jorge Silva. 2017. Transfer Learning for Education Data. *KDD Workshop* (2017).

[12] Ian T Jolliffe. 1986. Principal component analysis and factor analysis. In *Principal component analysis*. Springer, Kent, England, UK, 115–128.

[13] Marius Kloft, Felix Stiehler, Zhilin Zheng, and Niels Pinkwart. 2014. Predicting MOOC dropout over weeks using machine learning methods. In *Proceedings of the EMNLP 2014 Workshop on Analysis of Large Scale Social Interaction in MOOCs*. Association for Computational Linguistics, Doha, Qatar, 60–65.

[14] Christopher Vu Le, Zachary A. Pardos, Samuel D. Meyer, and Rachel Thorp. 2018. Communication at Scale in a MOOC Using Predictive Engagement Analytics. In *Artificial Intelligence in Education - 19th International Conference, AIED 2018, London, UK, June 27-30, 2018, Proceedings, Part I*. IJAIED, Berkeley, CA, USA, 239–252. https://doi.org/10.1007/978-3-319-93843-1_18

[15] Grégoire Mesnil, Yann Dauphin, Xavier Glorot, Salah Rifai, Yoshua Bengio, Ian Goodfellow, Erick Lavoie, Xavier Muller, Guillaume Desjardins, David Warde-Farley, Pascal Vincent, Aaron Courville, and James Bergstra. 2012. Unsupervised and Transfer Learning Challenge: a Deep Learning Approach. In *Proceedings of ICML Workshop on Unsupervised and Transfer Learning (Proceedings of Machine Learning Research)*, Isabelle Guyon, Gideon Dror, Vincent Lemaire, Graham Taylor, and Daniel Silver (Eds.), Vol. 27. PMLR, Bellevue, Washington, USA, 97–110.

[16] Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering* 22, 10 (2010), 1345–1359.

[17] Baochen Sun, Jiashi Feng, and Kate Saenko. 2016. Return of frustratingly easy domain adaptation. In *AAAI*, Vol. 6. Palo Alto, CA, USA, 8 pages.

[18] Baochen Sun and Kate Saenko. 2016. Deep coral: Correlation alignment for deep domain adaptation. In *European Conference on Computer Vision*. Springer, Springer International Publishing, Cambridge, MA, USA, 443–450.

[19] Kalyan Veeramachaneni, Una-May O'Reilly, and Colin Taylor. 2014. Towards Feature Engineering at Scale for Data from Massive Open Online Courses. *CoRR* abs/1407.5238 (2014).


## APPENDIX

## A MODEL ARCHITECTURES

The detailed architectures of the three neural network models used by all the transfer methods and the label-truth baseline are:

**LSTM Predictive Model** Input–Conv1D(16, 1)–ReLU –Conv1D(8, 1)–ReLU–LSTM(8)–ReLU–Flatten–FC(1)–Sigmoid –Output

**LSTM AE** Input–Conv1D(12, 1)–LeakyRelu(0.2)–BLSTM(8) –LeakyRelu(0.2)–Conv1D(8, 1)–Flatten–Embedding–Reshape –BLSTM(6)–LeakyRelu(0.2)–Conv1D(13, 1)–Sigmoid–Output

**CNN Predictive Model on AE embeddings** Input –Conv1D(8, 3)–ReLU–Conv1D(8, 3)–ReLU–Flatten–FC(1) –Sigmoid–Output

Where FC(n) is a fully connect layer with $n$ neurons, Conv1D(n, k) is an 1D-convolutional layer with $n$ output channels and kernel size $k$, LSTM(n) is a LSTM layer with $n$ cells, LeakyReLU($\alpha$) is a leaky-ReLU activation, and BLSTM(n) is a bi-directional LSTM layer with $n$ cells.

## B EXPERIMENTAL CONFIGURATIONS

We implemented the models using *PyTorch* and *Keras* in Python. For all training, the batch size is 128, the *Adam* optimizer is used with learning rate 0.001, and the number of epochs is between 100 and 200. For all AEs, the bottleneck size is eight dimensions per time unit. For **Label-Truth** and **Label-Truth-AE**, the train-test split ratio is 4 : 1. For **Passive-AE Transfer**, the number of output components of T-PCA is set to six per time unit. For **Active-AE Transfer**, the loss weights of the source prediction cross-entropy loss, the target autoencoding mean-squared error and the CORAL loss are 0.008, 1 and 1000 respectively. For each combination of training method, model architecture, source, target, and week we only train once, since all of our results are averages with low variance, and the variance from stochastic optimization here is very small (less than 1% according to our experiments).