Deep Queue Learning

Avoy Datta
Department of EE
Stanford University
avoy.datta@stanford.edu

Dian Ang Yap
Department of EE
Stanford University
dayap@stanford.edu

Zheng Yan
Department of CS
Stanford University
yzh@stanford.edu

Abstract

Office hours at Stanford are typically subject to significant variance in student demand. To tackle this problem, we predict student demand at any office hours on an hourly basis using data scraped from Queuestatus, Carta, and course syllabi. We conducted experiments using regression on fully connected NNs, univariate and multivariate LSTMs, and compared with an ensemble of multimodal classification models such as random forests and SVMs. We compared different losses such as MSE, MAE, Huber, and our own sqHuber against normalized inputs, and evaluate on student demand with and without smoothing. Results show that our models predict demand well on held-out test quarters both in seen and unseen courses. Our model could thus be a useful reference for both new and existing courses.

1 Introduction

Among CS students at Stanford, the experience of queueing at office hours (OHs) is practically universal. Office hours is an important part of any class, allowing students to get valuable one-on-one help. Unfortunately, student demand is prone to high variance, resulting in sometimes students waiting hours before receiving help, or conversely, teaching assistants (TAs) waiting hours before any students arrive. In particular, periods of overcrowding are a source of stress for both students and TAs, and are among the most commonly cited sources of negative experience on Carta. Thus, improvements in OH scheduling could significantly improve overall course experience for all parties.

However, as with all logistical decision making at universities, there are significant complexities in the process. Our project addresses the arguably most variable component of the input — predicting peaks of student demand. Using hourly OH data scraped from Queuestatus, course information from Carta, and major dates from class syllabi, we trained a fully connected neural network model that predicts the hourly load influx for any given course and quarter. We define the load influx as the average serve time for the day times the number of student signups. Conceptually, this is the aggregate TA time needed to satisfy all student demand over some period. Note: In terms of dataset and big-picture goals, this is a shared project between CS229 and CS221. For CS229, we focused on a more theoretical approach in predicting load influx by designing and evaluating new loss functions catered towards data with high variance and fluctuations. We also combine an ensemble of approaches to fine tune our prediction by using signal processing practices, as well as experiment with multimodal classification using SVMs and random forest models. For CS221, we focus on assigning TAs to the surge timings using modified Gibbs Sampling and EM algorithms, as well as LSTM prediction models.

2 Related Work

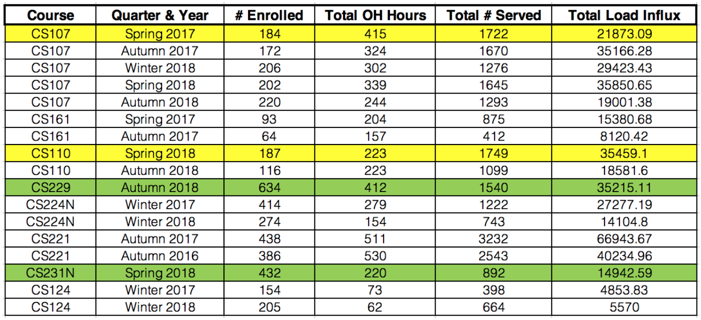
Unfortunately, predictions for time-series behavior of students are not very well studied. In an interview with CS109 lecturer Chris Piech, he expressed that status quo OH scheduling is somewhat arbitrary (loosely based on assignment dates). As a first source for technical work, we looked to the

CS229, Machine Learning. Autumn 2018.

approaches of a CS229 project that had a similar goal. Troccoli et. al used custom feature extractors to predict wait times at the LaIR (CS106 office hours) [1]. Interestingly, multimodal classification with equi-depth buckets outperformed regression approaches for them, indicating that a classification problem might be fruitful for our project as well. Fortunately for us, QueueStatus eclipses the LaIR framework in number of courses served and thus data collected, which allows us to build more generalizable models. Chatfield's work on statistical approaches to time series data also served as a useful source [2]. Chatfield recommended several transformations for time series data, included logarithmic transformations to stabilize variance and convolution as a smoothing method. We also noticed that our dataset contains significant outliers, and thus may be prone to overfitting them. To deal with this, we expanded the work by Huber who derived Huber loss used in robust estimation, where outliers are penalized less heavily than mean squared error loss [3]. However, the Huber loss is not differentiable everywhere, which could introduce complexities during backpropagation. Hanning's published self-convolution windows were also used effectively to smooth out harmonic data, which we referred to as another solution [4]. We also refer to Hagan's work on neural network architecture as a starting point for the relative number of neurons in our own models [5].

3 Datasets and Features

To obtain data, we set up a pipeline that scrapes hourly office hours from Queuestatus. Through customized JSON parsing, we were able to obtain a combined 17 quarters' of data across 7 prominent CS classes. After preprocessing to remove all entries with zero serves and signups, we ended with 4672 hours', or just under 200 straight days' worth of OH data. A summary is shown below.



Course	Quarter & Year	# Enrolled	Total OH Hours	Total # Served	Total Load Influx
CS107	Spring 2017	184	415	1722	21873.09
CS107	Autumn 2017	172	324	1670	35166.28
CS107	Winter 2018	206	302	1276	29423.43
CS107	Spring 2018	202	339	1645	35850.65
CS107	Autumn 2018	220	244	1293	19001.38
CS161	Spring 2017	93	204	875	15380.68
CS161	Autumn 2017	64	157	412	8120.42
CS110	Spring 2018	187	223	1749	35459.1
CS110	Autumn 2018	116	223	1099	18581.6
CS229	Autumn 2018	634	412	1540	35215.11
CS224N	Winter 2017	414	279	1222	27277.19
CS224N	Winter 2018	274	154	743	14104.8
CS221	Autumn 2017	438	511	3232	66943.67
CS221	Autumn 2016	386	530	2543	40234.96
CS231N	Spring 2018	432	220	892	14942.59
CS124	Winter 2017	154	73	398	4853.83
CS124	Winter 2018	205	62	664	5570

Figure 1: Full dataset. White: Train. Yellow: Test(Seen courses). Green: Test(Unseen courses)

We experimented with a plethora of features to augment our dataset with, and decided on the following predictors based on a combination of logic and significant correlation with load influx. On a per-class basis, we used: number of enrolled students, instructor rating, and proportion of freshman/graduate/PhD students enrolled. On a per-hour/day basis, we used: days until next assignment due, days after previous assignment due, days until an exam, hour of day, weekdays. For the hourly/daily features, validation testing found that one-hot bucket encodings were more effective for predictions. Day differences were bucketed in ranges of 10 to 5, 4 to 3, 2 to 1, and 0. Hour of day was evenly bucketed into morning, noon, afternoon, and evening. Each entry corresponds to one hour of OH, and every entry in the same course/quarter shares the same course/quarter features. As discussed later, we also experimented with log-transformations.

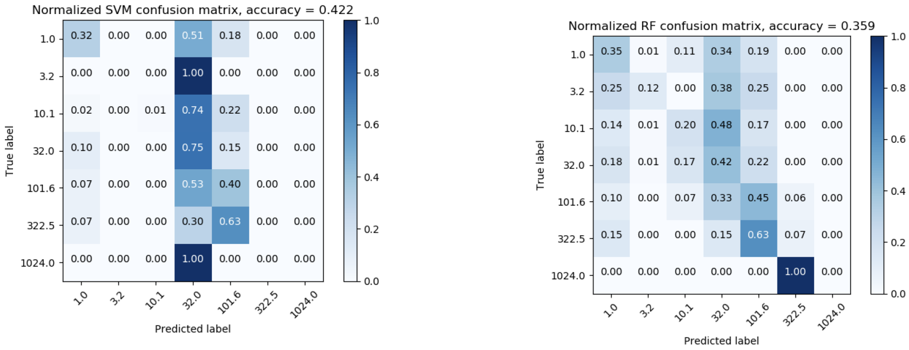
As our ultimate goal is to predict entire unseen quarters, we separated our training/validation/test sets by entire quarters. Due to our limited sample size, we use K-fold cross validation to tune hyperparameters, where K is our number of quarters. Our test set consisted of 4 total classes: CS110 Spring 2018 and CS107 Spring 2017 as unseen quarters of classes we trained on, and CS224N Winter 2018 and CS231N Spring 2018 as entirely unseen courses. Our training set thus consisted of the remaining classes, totaling 13 quarters' of data between 5 unique classes.

We note that after training models to predict load influx on these datasets, we do not predict hourly student demand for TAs, as is ideal. Rather, we predict hourly student demand for TAs given that office hours is held. We determined that current TA assignments are uncorrelated with time of day (p = 0.63, cor.test in R) and typically scheduled throughout active hours. Therefore, we assume that the status quo scheduling of office hours is frequent and unbiased enough such that real student demand is proportional to the student demand given office hours is held.

4 Methods

4.1 Multimodal classification

We first implemented multimodal classification models as baselines, where instead of using equi-depth buckets, we divided the minimum and maximum load influx into 7 logarithmic time buckets. Using SVMs with radial kernel and random forests with 1000 estimators, we obtained an initial baseline with accuracy 0.422 and 0.359 respectively, with the confusion matrix as shown below.



(a) Normalized confusion matrix of radial kernel SVM, $c = 1$

(b) Normalized confusion matrix of random forest.

Figure 2: Plot of confusion matrix of two classification models.

We see that even with fine-tuning of hyperparameters, the classification models have decent performance but with large skew and variance in predicting high load influxes, which could be possibly due to class imbalance in different buckets when on a log scale. We thus choose to focus on regression next to predict the spikes of load influx in different hours.

4.2 Regression: FCN, Huber and sqHuber loss

We also set up baselines by training fully connected networks and LSTMs for regression tasks. The FCN, which approximates functions with non-linearities and multiple layers that activate depending on weights mapped to higher dimension across stacked layers, has input size 30 (with our 30 features) with 2 hidden layers of size 15 and 4 respectively, followed by a single output final layer. Each hidden layer uses a ReLU activation function, with a linear activation for the output layer. We also experimented with 3/4 hidden layers which led to overfitting, even with normalization techniques that performed worse on the validation set.

LSTMs (Long short-term memory) is a form of recurrent neural network focused in 221 report). It addresses vanishing gradients while factoring in previous states with a recurrence formula at each time step, which makes it suitable for temporal data. We used two LSTM cells in autoregressive LSTM with window size of 16, and each output was fed back as part of the next window. All input features were normalized in a range $[0, 1]$ for every experiment, and all baseline models were compiled with Adam optimizer with early stopping to prevent overfitting. Due to insufficient data, we face high variance in training LSTMs with the initial baselines reported below.

Therefore, we choose to continue work on the fully-connected network (FCN). However, in our FCN, we notice our predictions for load influx throughout the quarter suffer from a consistent offset from the mean of the distribution. Upon inspection, we suspect that the large amount of outliers may have caused the bias due to their huge penalties while minimizing the L2-norm loss function. Thus, we seek a new loss function that doesn't penalize outliers as heavily. The **Huber loss** is particularly

Table 1: Evaluation of initial baselines and model choices.

Baseline Model	RMSE
FCN (Fully Connected Network)	**109.28**
FCN with Dropout	111.3
FCN with Batchnorm	125.7
Autoregressive LSTM	128.1
seq2seq LSTM	109.5

useful for this since it scales linearly outside a specified domain:

$$L(y,\hat{y}) = \begin{cases} \frac{1}{2}(y-\hat{y})^2 & \text{, if } y-\hat{y} <= \delta \\ \delta(|y-\hat{y}| - \frac{1}{2}\delta) & \text{, if } y-\hat{y} > \delta \end{cases}$$

We compare this traditional loss function with a novel loss function we designed for the purposes of experimentation- the **sqHuber Loss**. The sqHuber loss is defined as:

$$L(y,\hat{y}) = \begin{cases} \frac{1}{2}(y-\hat{y})^2 & \text{, if } y-\hat{y} <= \delta \\ \sqrt{\delta(|y-\hat{y}| - \frac{1}{2}\delta)} + (\frac{1}{2}\delta^2 - \frac{1}{\sqrt{2}}\delta) & \text{, if } y-\hat{y} > \delta \end{cases}$$

The sqHuber loss is piece-wise continuous, and scales proportional to the square root of the residual for values above a specified domain. Thus, it is even more robust to significant amounts of outliers.

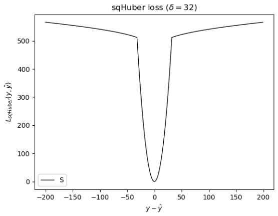


Figure 3: sqHuber Loss

4.3 Transforming the load influx data

The load influx is an erratic function. Large fluctuations, or 'spikes', in the load are difficult to predict without overfitting the model, thus transforming the training labels (actual load influx before training may be fruitful. We attempted two methods to transform our data for better predictions:

1. Hanning window: A 1-D convolution with a Hanning window. This reduces spikes and thus potential to overfit. [4]
2. Logarithmically scaling the load influx values in the training set, as student demand may be better represented as a geometric function of the inputs and this helps stabilize the variance.

5 Experiments

5.1 k-fold cross-validation

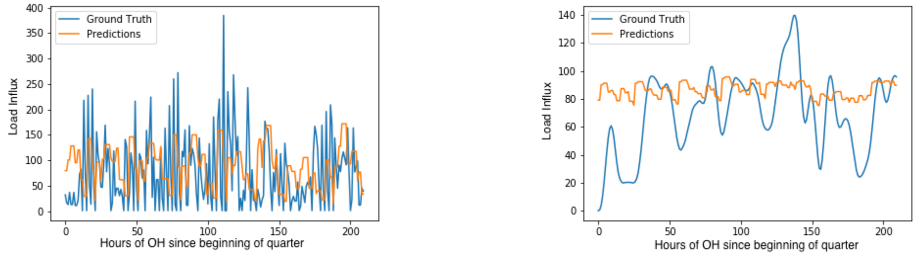
During validation tests, we find the root-mean-squared-error (RMSE) to be heavily dependent on the course and quarter used. Thus, to reduce variance in RMSE obtained from validation tests, we used **leave-one-out** cross-validation for our validation studies, where we stochastically choose a (course, quarter) pair as the validation set, and use the remainder of the **joint** training & validation sets for training. This process is repeated for $k = 8$ iterations, with the validation RMSE the mean of the results. Note that for each of the k iterations, the validation set was stochastically chosen and isolated from the training data, with the parameters of the model reset between iterations. The results for cross-validation between models are tabulated in **Table 1**, which was done preliminarily to select a model. Cross-validation results between data transformation methods is shown below.

Table 2: Comparison of **mean** validation set RMSE for different loss functions and transformations

Type of smoothing or loss function used	Huber	sqHuber	Mean squared error	Mean absolute error
No transform	109.47	126.20	**109.28**	110.55
Log transform	118.32	126.07	120.07	**106.11**
Hanning smoothing	107.43	**102.96**	122.78	119.70

5.2 Analysis of validation experiments, final model selection, and test results

Based on the analysis of our tests, we see the two best performing datasets are the sqHuber loss paired with windowing, and the MAE paired with log transforms. We performed validation on 8 quarters of data for each. For comparison, we arbitrarily picked out a dataset shared between the two.



(a) Trained on logarithmic scale of data with MAE

(b) Hann-smoothed with sqHuber

Figure 4: Prediction vs. ground truths for CS107 Autumn 2018.

Even though the sqHuber loss does marginally better (RMSE-wise) than the MAE with smoothing applied, training on log transformed influx allows the model to predict the general trends of student demand much better. Smoothing with a Hann window tends to even out our predictions, which although leads to lower error, removes much information desirable to instructors. Thus, we choose to proceed forward with the mean absolute error loss, with log transformation applied during training.

Final evaluation on test set. We obtained an avg. RMSE of 124.466 for our set of seen courses in an unseen quarter, and 106.478 for our set of unseen courses in unseen quarters. Furthermore, similar to Figure 4a), the relative locations of spikes were well captured in all four classes; however, in our unseen courses set, the magnitudes of the spikes were almost double in size of the spikes of the ground truth. From this, we see that our model remains relatively robust between both seen and unseen courses in terms of spike location and overall accuracy, but not spike magnitude.

6 Conclusion and Future Works

Overall, our project provides the first general-use model for predicting student demand at Stanford CS office hours. Using hourly Queuestatus data and course information, we were able to generate realistic predictions for office hours load in a wide range of CS classes. Ultimately, out of several tried, our best model was our fully-connected neural network, using mean absolute error and trained on log-transformed load influx. Although a slightly different model using our custom sqHuber loss gave marginally lower RMSE, it failed to retain spike information due to perhaps too much outlier penalty. Our RMSE indicates that the model is off by an average of 2 hours * students in testing. Empirically, we see this is a mostly a result of slightly misplaced and/or incorrectly heighted spikes. Since our final log model makes predictions that are then exponentiated, it often predicts the locations of spikes correctly, but fails to capture exact magnitude. Thus, although our system may not be able to predict exact student demand, it can still serve as a valuable guideline regarding when to expect relative peaks. Furthermore, we constructed a basic GUI in R that, given basic course information, generates OH hourly load influx for the whole quarter within a minute (demoed during poster session). So far, Chris Piech has expressed interest in using our model next spring. Given more time, we would like to extend our predictions to more classes, and perhaps even other universities using Queuestatus.

7 Contributions

All team members contribute equally to this project in terms of time and efforts.

1. Zheng focused on feature engineering and creating baselines for multimodal classification. All the dataset had to be manually scraped and combined from Queuestatus, Carta and syllabus pages of different class and Zheng was responsible for all the working data. Also worked on modified Gibbs sampling for 221 and responsible for feeding team members with good food.
2. Dian Ang worked on an initial proposal of sqHuber loss with less interesting results due to shift offsets and discontinuities. Also worked with an ensemble of methods to prevent model from overfitting, along with fine tuning of hyperparameters.
3. Avoy coined an improved version of sqHuber loss that addresses the shift offset. Besides building the multivariate LSTM, Avoy worked on k-fold validations, logarithmic scaling and Hann window smoothing. Also worked on modified Gibbs sampling for 221.

8 CS221 Final Report

Since the deadline for CS221 final report is after the CS229 final, we attach the link below where instructors are free to read through our CS221 final report as well.

https://github.com/Zheng261/DeepQueueLearning

9 References

[1] Troccoli, N., Capoor, B. & Troute, M. (2017) Predicting CS106 Office Hours Queueing Times. *Past CS229 Project*

[2] Chatfield, C. (2016). The analysis of time series: an introduction. CRC press.

[3] Huber, Peter J. Robust Estimation of a Location Parameter. Ann. Math. Statist. 35 (1964), no. 1, 73–101. doi:10.1214/aoms/1177703732. https://projecteuclid.org/euclid.aoms/1177703732

[4] Wen, H., Teng, Z., Guo, S. et al. Hanning self-convolution window and its application to harmonic analysis. Sci. China Ser. E-Technol. Sci. (2009) 52: 467. https://doi.org/10.1007/s11431-008-0356-6

[5]. Hagan, M. T., Demuth, H. B., Beale, M. H., De Jesús, O. (1996). *Neural network design (Vol. 20). Boston: Pws Pub..*