Title: Learning Average Course Load Hours from Student Responses

Group Members
Rujul Singh
Nick Romig
Kitty Moy
Noelle Jung

Introduction

One of the most pressing issues that college students face today is the issue of course load difficulty. At Brown, we are lucky to have an enormous amount of data (via the Critical Review) that allows students to make informed decisions on which courses to take.

Unfortunately, this kind of information is not available to students at many other colleges. Even at Brown, there are many Critical Review feedback forms filled out that don't contain much of the relevant information that students need to make course selection decisions (most specifically, the average hours per week). These reviews may include textual data (i.e responses to open-ended questions) and selected numerical data (i.e professor ratings) but do not include the statistics most relevant to college students today (hours per week).

We felt that this issue is something that could potentially be solved via Deep Learning methods. For those feedback forms that do not contain this salient information (average hours per week), we may be able to glean out the relevant numerical statistics based purely off of what is provided. Taking things one step further, this information could be used by university administrators at other colleges who have access to student feedback forms to accurately determine the workloads of different classes. Finally, for students, the text and numerical data obtained from sites such as "Rate My Professor" (which do not contain metrics for average hours of work per week) could be fed into this model as well to obtain this information.

Fundamentally, this problem encompasses a combination of NLP, numerical data analysis, and regression. We need to find some way to encode the information obtained from the text sources in the Critical Review Feedback forms, and then run a regression to calculate the statistics we need.

Methodology

We implemented three separate multi-channel multi-input models, all of which incorporated Natural Language Processing and numerous connected layers. These models were all multi-input models (received several disparate sets of inputs), but varied in the model architectures and inputs considered. Loss was calculated via the mean squared error methodology.

Base RNN Model
This model only used the text answers to the following three questions as input:

1. What types of assignments did this course have? Please check the appropriate boxes and provide the number/frequency/any other relevant information about each type of course work
2. Is there anything else prospective students should know about this course?
3. Discuss the instructor's teaching style. What was effective and what was not?

The responses to these questions were tokenized, and were lifted into an embedding space. Each word was represented by a 100 dimensional embedding vector.

Following the lifting of the textual answers, three separate Gated Recurrent Units (with an output size of 100) were used. Each response was divided into a “Window Size” of 125 (any shorter sentences were appended with padding). Three separate GRUs were used, as the contextual information provided by the text responses are likely to be different because they are responses to different questions.

Once the sentences were processed through the Gated Recurrent Units, the resultant vectors were processed through 2 dense layers of output size 100 each with a ReLu activation function. Each sentence had its own set of 2 dense layers.

Once each sentence was processed through their respective GRUs and dense layers, the resultant three output vectors were concatenated. This concatenation was then processed through a single dense layer of output size 50 and a ReLu activation function, then a final regression dense layer of output size 1 and no activation function. This final output represents the model’s predicted “average hours per week” for the course.

RNN with Dropout
This model was identical to the base RNN model, but included a dropout layer. Once the sentences were lifted into the embedding space, a dropout layer with a dropout rate of 0.1 was used on each individual embedded sentence matrix.

Multi-Input RNN integrated with Numerical Data
This was the most complex model that incorporated the most information available. Both numerical and natural language data were combined to create an integrated end-to-end regressor that integrated the predictive abilities of all attributes included in the dataset.

The natural language processing prior to the concatenation step in this model was identical to the prior two model architectures.

In addition to the natural language input, inputs for 15 separate numerical fields were also included in this model. These inputs were answers to the following questions:

1. What fraction of classes did you attend? (0-1)
2. Course Assessment - Assignments (readings, projects, homework, papers) were worthwhile (1-5)
3. Course Assessment - Class materials (slides, notes) were useful (1-5)
4. Course Assessment - Course was difficult (1-5)
5. Course Assessment - I learned a lot in this course (1-5)
6. Course Assessment - I enjoyed this course (1-5)
7. Course Assessment - Grading was timely (1-5)
8. Course Assessment - Grading policy was clear and fair (1-5)
9. Course Assessment - Recommended to non-concentrators (1-5)
10. Instructor Assessment - Presented material effectively (1-5)
11. Instructor Assessment - Used class time efficiently (1-5)
12. Instructor Assessment - Encouraged questions and discussion (1-5)
13. Instructor Assessment - Passionate about material (1-5)
14. Instructor Assessment - Receptive to student needs (1-5)
15. Instructor Assessment - Feedback was available and useful (1-5)

The vector of numerical inputs was independently processed through 2 separate dense layers (of output size 100 and 50 respectively), each using ReLu activation functions.

Once the numerical inputs had been processed through their dense layers, they were concatenated with three separate text inputs after these three inputs had been processed through their respective Gated Recurrent Units and two dense layers. This concatenated output was then processed through a dense layer (output size 50) with ReLu activation, and then finally through a regression layer (output size 1) which had no activation function.

Results

The model was unable to achieve the base accuracy goals set out in the previous check-in, but nevertheless still achieved significant improvements in accuracy after being trained. The range for average hours in the Critical Review Dataset was from 0 to 35 hours. With random guessing, this yields a MSE of over 200. The results from the three models (with all hyperparameters - epochs, batch size, learning rate - tuned to their optimal values) are detailed in the table below.

Model	MSE
RNN	14.92
RNN with Dropout	14.96
Multi-input Model	14.38

Based on the results, incorporating dropout in the model did not appear to have any drastic effects on accuracy. As expected, incorporating additional numerical information did lead to a noticeable improvement in accuracy (MSE dropped by around 0.6). We did expect, however, to see a larger improvement in accuracy via incorporation of numerical attributes. These results suggest that text based context contained within the open-response questions are much better indicators of average workload compared to the numerical responses.

Challenges

We were faced with numerous challenges throughout the process of model implementation, the most significant of which are detailed below.

1. We needed to determine a method to handle mixed data (combination of numerical and language based data) and multiple inputs, both of which have not been encountered in class before. This required us to research and implement parallel pipelines of neural nets to handle each input and data type, and determine a method to combine these networks into a single output.
2. There were many inconsistencies within the text data. Punctuation and capitalization were inconsistent, there were some spelling errors, and overall the data was not very clean. We assume that many of the people filling out the form are filling it out haphazardly/not making sure that all of their spelling and punctuation is 100% correct. To counter this issue we needed to handle dozens of edge cases dealing with punctuation and word splitting to ensure that all the sentences were properly tokenized.
3. There was a lack of data in cleaned CSV format. Originally, we only had a little over a thousand data points. This was not enough for the model to learn properly on, and MSE was much higher. We needed to dig through Critical Archives from the past 5 years and scrub all data into one consolidated CSV file to increase the amount of data that our model had access to to train from. Eventually, we were able to increase the size of our dataset to just under 3,000 individual data points.
4. The model was very sensitive to hyperparameter choices. Hyperparameters (i.e epochs and learning rates) that were used for the first two models were not yielding great results for the third model. Our model was computationally very intensive to train given the sheer number of hyperparameters, and so we needed to utilize a grid-based search pattern and run the model for many hours to try to find the optimal set of parameters.

Reflection

We feel like our project turned out well. The mean squared error of our results is significantly lower than just randomly guessing the number of hours ranging between 0 and 35. An error of approximately 3.8 hours is ultimately not too bad in approximating the number of hours needed for a class over a week. Our mean square error, however, did not reach our base/target/stretch goals of less than 5, 3, and 1 hour respectively. We originally thought that we would have more data, however, we lost a lot of responses after cleaning out the data that did not contain all of

our necessary parameters. With more data points, we believe that our mean square error could be further decreased.

Additionally, we were a bit surprised that there was not a substantial difference in error with the addition of the numerical data, but that is probably due to the fact that the multi-input RNN would require different hyperparameters. We were hesitant to drastically change the hyperparameters from method to method, as we wanted a relatively consistent baseline to compare the models against and were concerned that we would be able to find the optimum set of hyperparameters for one model, but not the others. Further optimization of hyperparameters for our numerical model could improve the MSE. Only the learning rate was adjusted compared to the other models to prevent overfitting, as we had much more information being passed through.

Our original plan for our model was using a transformer, as transformers typically perform better than their RNN counterparts, however, we struggled to come up with an idea using the transformer model that incorporated the different types of responses (ie the different text responses to each question, and for the multi-input RNN, with the numerical ratings of each category), so we switched over to a RNN model.

Improvements and Next Steps

If we were to redo our project, there are a few changes that we would make to our approach. The largest improvement we could make would likely be to include more review data in our data set, since we believe that the fact we only had about 3,000 reviews limited the capabilities of our model. We found that after training for too many epochs, the model started to exhibit characteristics of overfitting, specifically getting high accuracy on our training data but failing to improve on the testing data. If we had more data, we could train our model for longer without risking overfitting, because the model would not be able to adapt to the dataset. Unfortunately, we already exhausted the Critical Review's set of digitized data, so if we were to get more data it would have to come from scans of old reviews from before the Critical Review was brought online. Translating these scans into trainable data would either need the implementation of an entirely new deep learning algorithm to turn scans into digital text, or an extremely large amount of man hours manually entering the data. This will likely be a step we look into if this model is used in practice by the Critical Review.

Additionally, it would make sense to attempt to build a model entirely using numerical data. If we could see that there was also significant predictive capacity in the numerical data, it would help explain the surprising result that including it in our model did not significantly improve our results. A good result using numerical data would suggest that either our model is not taking good advantage of this data, or it is doing so in a way that sacrifices some of the predictivity of the text data. If we find that there is little that can be surmised from the numerical data model, it could support the conclusion that the data says little about the hours per week a class requires. This seems hard to believe, however, because some inputs, like class difficulty, could be expected to correlate highly with hours spent per week.

Takeaways

We feel that the takeaway from our project is that the hours of work for a class can be deduced using other data about the class to a certain degree, allowing for some variance to do class-specific factors. Our project shows that there is enough information in the written responses provided by students to interpret the hourly commitment, and that there is some benefit to using the numerical data from the Critical Review as well. With further optimization of our hyperparameters and inclusion of more data, we believe we can improve these predictions to attain further accuracy. We are pleased that we were able to implement a model that combined multiple RNNs for text data usage and a neural network for numerical data, something that we were not able to do in class.

This model will hopefully be of much use the Critical Review, and further students at Brown. As we saw from the fact that we had to omit many examples, and adapt some incomplete data for better use in our model, the problem of incomplete data is a large one for the Critical Review. It is not hard to imagine a future, not too far away, where courses that have significant text data can still have accurate predictions for the number or hours required per week, which would allow students to more easily plan their semesters. It is possible even that further information could be deduced from bits and pieces of student reviews. Overall, we believe that our model creates the opportunity for much more information about potential classes to be communicated to students.