Background Literature

As some background for this project, the Gernsbacher lab provided us with some scholarly articles that serve as the foundation for the current questions they are trying to answer. Professional development (PD) and its iterative process is the main topic of interest here, and more specifically, professional development in the sphere of higher education. Many universities develop some form of in-person support centers where instructors, usually of any discipline, can visit and receive guidance to further their skills in instruction (Jacob et al., 2015). These centers can provide resources that range from general work-life balance to extremely specialized instructions on how to approach teaching a course (Pesce, 2015). While these centers are slightly contentious in their practical use, Jacob et al. (2019) highlight some of the most impactful strategies that these centers can offer that would greatly improve their perceived usefulness and efficiency. For one, the presence of such resources greatly reduces the stress for professors that are responsible for teaching and research by making the teaching side of their jobs much more constructive and not feel as much like a burden. These centers should also be rewarding excellence in the realm of teaching and research which creates a much more positive cycle within the institution. By having these institutions be supportive of a professor's research interests and understanding their primary/secondary role as an instructor, the center can much more effectively support professor's by pushing them to incorporate their research into the course whenever it can be applicable. While these centers are extremely valuable resources for institutions to implement, there are definitely downsides and barriers to use that are present. So how can we tackle some of these barriers? A solution that recent research has been really honing in on is utilizing social media as a platform for PD.

Social media as a platform has most certainly become a cornerstone of modern society and is greatly understood for its ease of accessibility in socialization and community building. Both of these aspects lend themselves greatly to aspects of PD that can be lacking in the traditional in-person model. For one, social media allows for asynchronous interaction in that posts made to a social media network at any point in space and time, and then consequently interacted with by an individual at any point in space and time. Also, networking with those in a similar field and creating large communities of those with a wide range of experience in a field is a great benefit of its utilization. These are just the tip of the iceberg when it comes to the positive impact of using social media for PD, but it is great in highlighting the potential for its use in higher education institutions. However, the main problem with getting this off the ground is convincing professors of these benefits.

While the opportunities of social media are great for PD, many are hesitant to hastily adopt it. Purvis et al. (2020) saw a great number of professors that were worried about blurring the line of authority and confidence in the validity of receiving help in such a manner. Many show concern of keeping a complete separation of personal life and professional life since utilizing social media for both would blur the line and could potentially lead to a heavy disruption in work-life balance. This is also given a solid understanding of social media use. Acuyo (2022) found that a very large hurdle in incorporating social media into PD was the faculty's lack of confidence in being able to capitalize on the benefits of social media use. While the professors were solidly behind the benefits presented to them, they were overwhelmed with worries of not being familiar enough with social media use to capitalize on these benefits. Even further, those that are familiar with social media use showed worries of not being able to make the jump from personal social media use to professional social media use. To wrap up this introduction on some of the background research, I just want to highlight that social media poses a great opportunity for PD but the biggest hurdle for implementing it is faculty support and consistent engagement with it. So going forward in this project, the primary question that the Gernsbacher lab is looking to answer is what individuals are going to social media for (in terms of PD) that differs from those going to traditional sources.

Project Goal (General)

The goal of the lab is to code Facebook posts for a number of different metrics. These posts come from the Society of Teaching Psychology Facebook page. This page is not under control of the lab or university and has also since been privatized since collection of this data occurred. Around 15,000 anonymized posts were collected from 2018-2023 along with some corresponding metadata. The coding scheme for these posts are separated into three separate prongs, Purpose, Discipline, and Identity of Poster. Purpose is fairly straightforward, what is the reason that they made this posts. This is first broken down into sharing vs requesting information. Then it is further broken down into more specific sub-labels that dive into more specifics about the kinds of information they are sharing or requesting. The discipline is definitely the weakest in terms of operationalization but it deals with whether or not the post is directly related to the field of psychology. Possible labels for this are Yes-Syllabus, Yes-Textbook, Yes-Job posting, and No. Finally, the poster identity prong deals with the type of position held by the individual making the post. Graduate student, K-12 Teacher, Non K-12 Professor, and Non-Instructor poster are possible codings.

Project Information and Specific ML Goal

The overall goal of this project is to create machine learning models that will help automate this process given the amount of time it takes for the team to code posts. We will be creating machine learning models that predict the label of the post (depending on the prong) using the text of a post (and potentially metadata!). The plan of action that we settled on after discussions with Tim, was to first run simpler classification models using GLMNET and Random Forest algorithms, then use Structured Topics Modeling (STM), and finally use OpenAI's backend API to access ChatGPT as a form of using a Large Language Model. For clarity, we will be running these "level" of models on each prong. Our full dataset was around

15,000 of the posts, but our training dataset was 600 posts and the test set was 1,000 posts. The training and test sets were coded by human coders.

Model Backgrounds

The level 1 models required a little less background research as they are the same kind of models that we learned about in 752, but we heavily drew on the textbook that was mentioned in his class by Hvitfeldt and Silge (2022). We primarily focused on using GLMNET and Tree-based models because each of the models would at some point require multi-level classification and these algorithms can (technically) handle them. We don't expect extremely good performance from these models but they are a nice starting point and give us much more interpretable results.

The level 2 models were definitely much more of a dive into unknown territory given that we did not cover unsupervised approaches in 752. Also, even though I have some background knowledge of how STMs work, I was entirely lost on using them as a predictive tool. In order to understand in more depth how Structured Topic Modeling works, we followed a tutorial and textbook from Valeria Hase (I've been trying to access this resource again, and it looks like she may be doing some remodeling of her projects? So in the references I have instead linked to her website that I hope will eventually include the resource!) To give a short run down of how these models work, they are an unsupervised approach that when you give it a (K) number of topics, it will find this number of topics within your documents. What is more useful for us in this project is the resulting topic probabilities for each post across each topic. These models are also great because they allow us to seamlessly integrate non-text data as part of the structure which means we can use our metadata to be incorporated here. Something important to note here is that the number of topics can be any number, it doesn't necessarily have to be equal to the number of possible labels.

Then, to actually get predictions, we employ K-means clustering analysis. This analysis will essentially create a K number of clusters, which in our case will be equivalent to the number of labels we are looking to assign for. The clustering will be assigned by looking at each observation and assigning a cluster based on posts most similar to itself. Then, in order to actually assess the performance of these topic/cluster formations, we would need to utilize a KNN model with our training data acting as a validation set.

Finally, the level 3 model is the backend API for OpenAI's ChatGPT. The exact model that we settled on was the 4o-mini model due to its cost efficiency for this project. It may be possible that another model may perform better, but for the sake of this project's budget and comparable performance, we settled on the 4o-mini. This seemed like it would be the most robust option of the models with the drawback of complete lack in interpretability. So we expected this to be our best performing level of models. In terms of background knowledge, this was the only level that required us to use Python but was simple enough that we did not have to do any extensive learning of new concepts. The biggest learning curve came in actually drafting up the most effective prompts for our prongs, but there was not a ton to learn leading up to this. A majority of what we did need to learn was provided in the form of tutorials on OpenAI's website and their documentation.

For my portion of the project, I was responsible for the Purpose prong and the Identity of Poster (and eventually handled the level 3 models for the Discipline prong).

Purpose

Level 1

Going into the level 1 Purpose models, it became clear extremely quickly that we had a severe issue in terms of class imbalances as can be seen in Figure 1.

Figure 1

Number of label observations in Request posts in the Training data set.

Label	N
Requests Materials	170
Requests Research	14
Requests Collaboration	10
Requests Advice	133
Requests Commiseration	10

These class imbalances are present throughout each prong and lead to a variety of issues. Our first step in trying to mitigate the effects of these was choosing the correct performance metric. Accuracy is a good general metric due to it's interpretability but it can be extremely misleading. The reason being that a model that does very well on accuracy may simply be over guessing the majority class label and doing good because of the small number of cases in the minority label. To this end, we decided to utilize an F1-Score, since it can take into account to a certain degree these imbalances and give us a bit clearer idea of how well our models are performing. So going forward, in every prong, we utilized accuracy, roc_auc, and an F1 score in order to assess model performance.

Another aspect that we thought to utilize prior to our model running was that the Purpose and Discipline prong both have a binary split then go further into specific multi-classification. On top of this, multi-classification (with class imbalances) is an extremely difficult task due to the complexity of it. So in order to reduce this complexity, we decided it would be best to first run binary versions of our models and use these predictions to simplify our multi-class prediction. We would want to use our best binary model and best multi-classification model in tandem so as we used more models, we decided to return to previous levels with better binary predictions.

An aspect of these posts that plays a large role in the human coding is the presence of URLs. In many cases of natural language processing, URLs are usually filtered out but we believed that how these are treated would be a crucial aspect to the model's success.To this end, a lot of the feature engineering involved different ways of handling these URLs. They ranged from making no change, replacing the URL with a placeholder (such as URL), extracting only the domain, or extracting the domain and article title. This is our biggest piece of feature engineering at this level and I will talk a bit more later about how it turned out.

One last distinction that I want to make is that we had quite a few discussions on the utilization of metadata. The lab was initially wanting us to utilize metadata given that the human coders were not making use of it in their coding. However, we believed that this inclusion would lead to better model performance so we would run all our models throughout the prongs both using and not using metadata to see if there was significant increase in performance.

The first model we ran was the binary split where we would simplify all our labels to be either Request or Shares, at this level and level 2 we did not include Uncodable type posts in our model building due to the complexity of them and relatively low frequency. The first major finding is that the inclusion of metadata leads to significant performance across the board. Our best no metadata was a Random Forest model that performed with 0.921 accuracy and an F1-Score of 0.900 whereas our best performing metadata model was a Random Forest model with 0.945 accuracy and an F1-Score of 0.945. (Important to note that these are our performances in the training/validation set and I will be reporting out performances in the test set in a separate section.)

Another crucial finding here that I will mention again in the multi-classification is that our feature engineering was not very successful in improving performance. We ran separate models that utilized differently engineered text posts based on the URLs as mentioned above. But we found that the full post with no modifications consistently performed the best of all the configurations. The best models mentioned above were both using the full post as opposed to one of the engineered versions of the post.

Overall, performance here is much better than expected and gives a great insight into how well we will be able to perform the binary into multi-classification pipeline later on.

For our multi-classification, out of curiosity of how bad it would perform, we first ran our same outline of models as mentioned above for all posts. This means that our classification was trying to predict out to all 12 possible labels that includes both Shares and Requests. The best performing model here was again a Random Forest model that performed pretty poorly overall with an accuracy of 0.601 and an F1-Score of 0.580.

Then we took our predictions from the binary classification and split our training data into only Share and requests posts and ran our models again on each type of post. This meant that we were simplifying the classification from 12 down to 7 and 5, respectively. For our request posts, our best performing model was a GLMNET with an accuracy of 0.683 and an F1-score of 0.682. For our share posts, our best performing model was a Random Forest model, BUT with the URL information being extracted. This model performed with an accuracy of 0.615 and a F1-Score of 0.683. While this is a definite improvement in performance, it may be a bit misleading as we will see later in the performance in the test set.

Level 2

After some discussions with John, we decided it would be a good idea to run a baseline model prior to running our STM models. This baseline would be simple classification models that would only be utilizing the metadata and no text data. This would give us an idea of how much benefit we are actually getting from utilizing the STM and their ability to incorporate metadata and contextual structure. Our best baseline binary model is GLMNET that has an accuracy of 0.912 and an F1-Score of 0.886. Our best multi-classification was a GLMNET that has an accuracy of 0.503 and an F1-Score of 0.498. What this tells us is that the metadata is extremely important in making the initial distinction between shares and requests, but it may be a little less important for distinguishing between the specific labels.

Then we moved on to running our binary classification. We decided that the feature engineering from our level 1 models were not very effective so we are abandoning this approach here on out. There is also quite a bit of data pre-processing that came along with this level given that we are doing an unsupervised model building into supervised classification for performance. There was also a lot more cleaning that had to be done because the full dataset had a lot more wonky posts that our STMs couldn’t handle so we just had to find those and pull them out one at a time. Within our STMs that utilized meta data, we created some features from our metadata alongside some common features that are included in these types of models. This included likes, comments, shares, year posted, month posted, day posted, link presence, and character count of the post.

When we actually were creating our model configurations, we had to do some improvisation on actually tuning for our hyperparameters. For our binary model, we tested on a range of 2 to 27 topics. Then we did K-means clustering for all our models created and ran a KNN algorithm to figure out which model configuration performed the best. Our best performing model was the meta-data STM with 4 topics that had an accuracy of 0.790 and an F1-Score of 0.843. Why do we see such a drastic decrease here? It may have something to do with the great performance we saw in the baseline. The metadata alone is great at predicting into the binary, so an STM might be having a hard time finding topics that fit both our textual data and our metadata into coherent topics.

This might be a bit of a spoiler, but our level 3 binary predictions were our best model for predictions, so those predictions are the ones that we used for our multi-classification feedforward pipeline. In order to capture the number of topics, we instead opted to search for topics in the range of 5 to 35. This could be broadened in the future, but to save time on computations this is the range that we stuck with. Our best request model was a 10 topic STM with metadata included that had an accuracy of 0.712 and an F1-Score of 0.737. Our best share model was a 17 topic STM with metadata included that had an accuracy of 0.682 and an F1-Score of 0.763. Something important to note here is that these best performing models were not run solely on the cluster analysis. In our previous models, we would run our KNN by regressing our true label on the cluster analysis results. In these best models, we found that including the model embeddings which we found as useful aspect of topic modeling in Kelechava's (2019) article. What this means is that our performance is best when we take into account both topic probabilities (after doing Principal Component Analysis) and our cluster analysis labels.

Level 3

Moving on to our final level of models for the Purpose prong, we are starting to use the OpenAI's backend API for ChatGPT. As we described in the Model Background section, the specific model that we settled on using is the 4o-mini model. Model building for this section is fairly straightforward given that we are utilizing a pre-built LLM, so most of the model building comes down to crafting a good prompt. The two approaches we started off with were a zeroshot prompting and fewshot prompting method. The zeroshot method is much more barebones as you are providing the labels in the prompt that are possible to give the post and the post itself. The fewshot approach requires more tuning as you are including an (or a few) examples of possible posts for that label. This gives the LLM more context to work off of and will generally perform better but is sometimes not necessary depending on the type of task.

For some general findings, we eventually stuck to only doing fewshot given that it was working the best and greatly reduced the amount of computation/expenses required to run the models. Another thing that we were constantly running into is that no matter how hard we emphasized in the prompt to only return one the possible labels that we wanted, ChatGPT would always return a few posts with labels that were not within our coding scheme. To save time, we simply removed these but in the future it may be best to actually look through each of these posts and apply an actual human label.

For our binary model, our fewshot model our accuracy overall was 0.964, overall F1-Score was 0.962, F1-Score on Requests was 0.971, and the F1-Score on Shares was 0.953. This is really great performance, and something that we can see that is interesting is that the

LLM is better at correctly identifying request posts as opposed to share posts. This is further supported by our request posts having a Precision score of 0.957 and our share posts having a Recall score of 0.931 which for both of these the other metric is a bit higher. Since this our best binary performance overall, we will be using these predictions to separate our dataset into shares and requests.

For our multi-classification, we first split the dataset based on our binary predictions and then created a prompt for our possible request labels and another for our possible share labels. Again, we only did this using a fewshot approach to save time and given that we did not believe the zeroshot would be any better. The accuracy of this model is 0.740 and the F1-Score is 0.745. This is somewhat comparable to our level 2 results, but definitely a bit higher. I also expect that when we get to assessing our model performance in test, the level 3 model will be much more consistent when it comes to generalizing to new data.

Model Performance in Test

In order to determine which level of model is our best model for using their predictions, we tested the best model from each level on the test dataset.

Binary

For our level 1 models, our best model performed with an accuracy of 0.941 and an F1-Score of 0.934. For our level 2 models, the baseline model performed with an accuracy of 0.894 and an F1-Score of 0.868. The STM model performed with an accuracy of 0.864 and an F1-Score of 0.831. For our level 3 model, we had an accuracy of 0.973 and an F1-Score of 0.972. So overall, our best performing binary model was the level 3 model that utilized ChatGPT.

Multi-classification

For our level 1 models, our macro (meaning no prior distinction between posts) model performed with an accuracy of 0.622 and an F1-Score of 0.522. The request model had an accuracy of 0.709 and an F1-Score of 0.730. The share model had an accuracy of 0.612 and an F1-Score of 0.354. Something very interesting to see here is how well our request model was able to generalize whereas how horrible our share model was able to generalize. For our level 2 models, the request model had an accuracy of 0.698 and an F1-Score of 0.711. The share model had an accuracy of 0.618 and an F1-Score of 0.651. For the level 3 model, we had an accuracy of 0.761 and an F1-Score of 0.763.

Purpose Prong Takeaways

The main conclusions we can take away from this prong is that we can predict very well in terms of binary predictions, but the multi-classification is much harder to do. We also found that overall, metadata definitely helps in building better models for this prong so that was a worthwhile aspect to look into. A big issue to address is the potential multicollinearity that is present in the specific labels. This definitely seems to be an apparent problem in the Share posts as we consistently have a harder time with the Share posts compared to the Request posts. There are two current suggestions I have moving forward for this prong. The first is to definitely experiment with more variety of prompts and model configurations using the OpenAI API. From my research into utilizing this API, there are a lot of much more complex model pipelines and embeddings that can be found using this so I think those would be a good avenue to look into. Next would be to look at how to deal with getting a more distinct separation between the specific labels. It definitely seems like some of the labels struggle from a bit of multicollinearity. This suggestion is definitely more on the Gernsbacher Lab side of things, and I don't think they are too keen on changing any labels at this point. So figuring out model configurations that can try to address these issues would be the best way to approach this.

Discipline

While I was not in charge of the Discipline prong for my part of the project, I did end up doing the level 3 model for discipline due to time constraints. So I will just be doing a quick run-down here. Currently, all I have done is results of doing a no metadata fewshot approach. In the validation set, it had an accuracy of 0.738 and an F1-Score of 0.795. In the test set, it had an accuracy of 0.685 and an F1-Score of 0.754. Something to note here, is that the performance drop is due to a drop in accuracy/F1 Score for the No cases. This led me to believe that the question we were posing in the prompt causes some amount of dissociation between ChatGPT's responses. I will be doing some final wrap-up work to make sure that we our deliverable for each prong ready for the Gernsbacher Lab.

Identity of Poster

Level 1

Overall, this prong was a bit more straightforward than the other prongs because there is no room for a binary into a multi-classification pipeline. The first thing that became very apparent when using our supervised model approaches was the severe class imbalances that are also present in this prong, but even more worrying was that one the most common observations was "Uncodable Poster." This label poses a great amount of difficulty for more traditional ML models given that there isn't as much structure to them. As a reminder, we had 600 training posts to build our models on, and since we had to remove these Uncodable Poster posts, we were left with only 397 posts to train on.

Figure 2

Number of label observations in Poster posts in the Training data set.

Label	N
Graduate Student	5
K-12 Teacher	4
Non K-12 Professor	236
Unspecified Teacher	147
Non-Instructor Poster	5
Uncodable Poster	203

Our best performing model in level 1 was a Random Forest model that included metadata. The accuracy for this model was 0.652 and an F1-Score of 0.631. This performance is not too great, but something interesting that we are able to extract from this model is what tokens are the most important in the model's decision making. These tokens included things like "https", "teaching", "course", "research", and "my" which are good signs since these are all related to the question of poster identity.

Level 2

In our level 2 models, we faced the same issue of dealing with Uncodable posts as well. On top of this, there were quite a large number of posts that were just not able to be used in the model training for the STM, as previously mentioned in the Purpose section. Our best model here was a 16 topic STM that utilized metadata and model embeddings with an accuracy of 0.605 and an F1-Score of 0.752.

Level 3

In our level 3 models, we wanted to try and address the issue of not being able to handle Uncodable Poster posts. To this end we included Uncodable Poster as a possible label for the model to use and included performance metrics with and without these posts. We only ran a no metadata version of this model because we did not think the metadata would significantly improve performance for this prong's question. This model had an accuracy of 0.45 and an F1-Score of 0.475. This is a terrible drop off in performance! This is very worrying given that the LLM is what we expected to always be our best performing model given its flexibility.

Model Performance in Test

For our level 1 model, our best model performed well in test, with an accuracy of 0.712 and an F1-Score of 0.705. This is quite surprising, given the complexity of our task and my expectation was that this performance would start off horribly and get better at each level. Granted it's with the caveat that Uncodable Poster posts were included, but nonetheless an interesting find. For our level 2 model, we have an accuracy of 0.541 and an F1-Score of 0.702. For our level 3 model, the accuracy of our model that included Uncodable Posters was 0.471 and had an F1-Score of 0.495. Without the Uncodable Posters, our performance slightly improves to 0.603 accuracy and an F1-Score of 0.598.

Identity of Poster Prong Takeaways

The biggest roadblock that needs to be addressed in this prong is the issue with Uncodable Posters. Our traditional (simpler) models can't incorporate them and even the flexible LLM can't precisely identify them. We may want to consider a pipeline here where we first ask ChatGPT whether or not we can identify the poster at all. This would be similar to the binary split in the Purpose prong. If we do this, then it may be possible to either continue using the LLM on only the posts that are able to be identifiable or use one of our other model types to actually produce predictions for the type of identification.

Another issue that needs to be dealt with is the multicollinearity of the labels. I think that the main reason that the LLM approach has difficulty outside of the Uncodable Posters is due to the overlap in label definitions. I also think that the issue is even more present here than in the Purpose prong, given how we are narrowed down so far in terms of scope of identity.

References

Acuyo, A. (2022). Reviewing the literature on professional development for higher education tutors in the work-from-home era: Is it time to reconsider the integration of social media?. *Education and Information Technologies, 27, 89-113.* https://doi.org/10.1007/s10639-021-10603-2

Hase, V. (2024). *Text as data methods in R.* https://valerie-hase.com/project/lehre_11/

- I'm not sure what is happening with this source at the moment, but it seems like the professor who made this document is revamping her course page? This is a resource that Tim gave us a while back when he went over STMs.

Hvitfeldt, E., & Silge, J. (2022). Classification. *Supervised Machine Learning for Test Analysis in R.* CRC Press. https://doi.org/10.1201/9781003093459 or https://smltar.com/

Jacob, J. W., Xiong, W., & Ye, H. (2015). Professional development programmes at world-class universities. *Palgrave Communications, 1*(15002), 1-27. http://dx.doi.org/10.1057/palcomms.2015.2

Jacob, J.W., Xiong, W., Ye, H., Wang, S., & Wang, X. (2019). Strategic best practices of flagship university professional development centers. *Professional Development in Education, 45*(5), 801-813. https://doi.org/10.1080/19415257.2018.1543722

Kelechava, M. (2019, Mar 3). *Using LDA topic models as a classification model input.* Medium. https://towardsdatascience.com/unsupervised-nlp-topic-models-as-a-supervised-learning-input-cf8ee9e5cf28

Pesce, J. R. (2015). *Professional Development for Teaching in Higher Education: Faculty Perceptions and Attitudes* [Doctoral dissertation, Boston College]. eScholarship@BC. http://hdl.handle.net/2345/bc-ir:104134

Purvis, A. J., Rodger, H. M., Beckingham, S. (2020). Experiences and perspectives of social media in learning and teaching in higher education. *International Journal of Educational Research Open, 1(100018).* https://doi.org/10.1016/j.ijedro.2020.100018

Sabbagh, R., & Ameri, F. (2020). A framework based on k-means clustering and topic modeling for analyzing unstructured manufacturing capability data. *Journal of Computing and Information Science in Engineering, 20*(011005), 1-13. http://dx.doi.org/10.1115/1.4044506