Academic Paper Acceptance and Category Prediction

Background

What is arXiv?

arXiv (pronounced "archive") is an open-access repository of electronic preprints, known as e-prints, of scientific papers in the fields of mathematics, physics, astronomy, computer science, quantitative biology, statistics, and quantitative finance. arXiv was created and is operated by the Cornell University Library.

arXiv is a popular platform for researchers to share their work with the scientific community before it has been peer-reviewed and published in a traditional journal. This allows researchers to get feedback on their work and allows others to build upon it more quickly.

Accepted and unaccepted papers in arXiv

arXiv is an open-access repository of electronic preprints, which means that it is a platform for researchers to share their work with the scientific community before it has been peer-reviewed and published in a traditional journal. As such, the papers on arXiv have not necessarily been accepted for publication.

arXiv does not perform any formal review or assessment of the papers that are submitted to it. Researchers are responsible for ensuring that their papers are of sufficient quality and meet the appropriate standards for their field before submitting them to arXiv. However, arXiv does have some basic guidelines that papers must follow in order to be posted on the platform, such as the requirement that papers be in English and be correctly formatted.

Paper Acceptance/Category Prediction and Challenges

It could potentially be helpful to be able to predict the acceptance rate of an academic paper, as this information could help researchers to understand the likelihood of their paper being accepted by a particular journal or conference. However, it is important to note that the

acceptance of an academic paper is influenced by many factors, and it is often difficult to predict with certainty whether a particular paper will be accepted or rejected.

Some of the factors that can influence the acceptance of a paper include the quality and originality of the research, the clarity and organization of the writing, the relevance of the work to the journal's focus and readership, and the overall fit with the journal's editorial goals and policies. These factors can be difficult to quantify and may vary widely across different journals and fields.)

In addition, the review process for academic papers is often subjective, with different reviewers having different standards and priorities. This can make it difficult to predict how a particular paper will be received.

Overall, while it might be possible to build a model that could predict the acceptance rate of an academic paper to some extent, it would likely be quite challenging to do so accurately and reliably.

Challenges faced and overcomed：

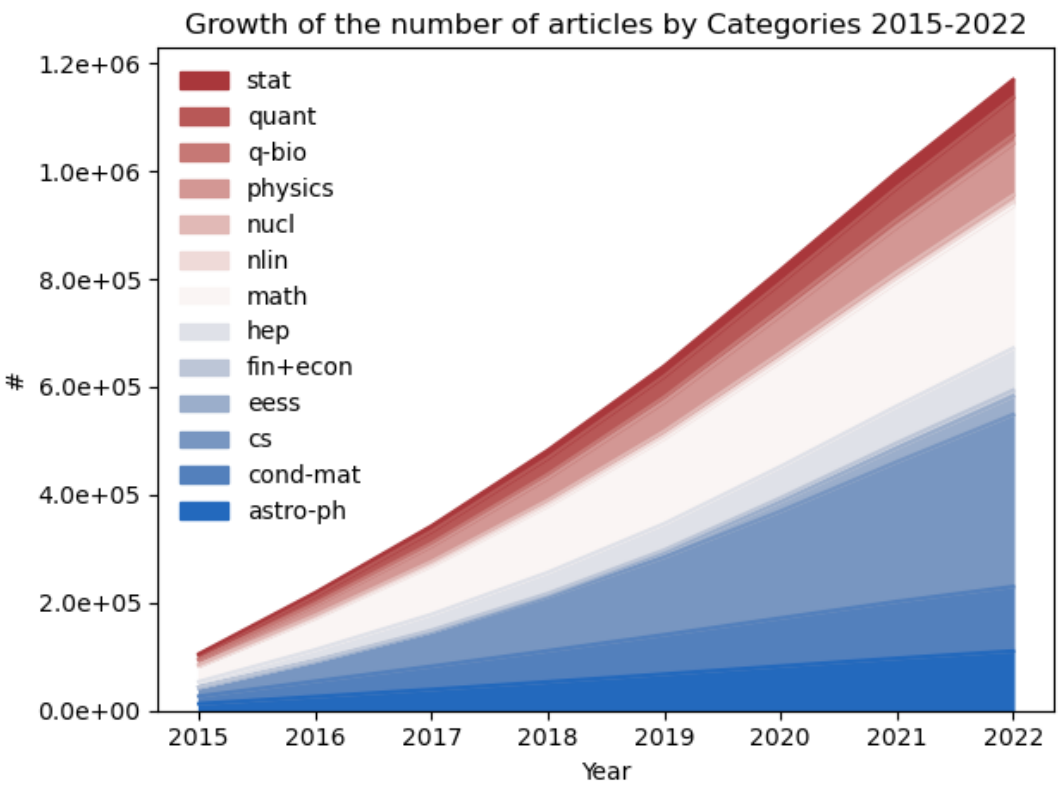
- Data visualization: choose accurate and correct data and graphs types to visualize data trends and evaluate the models.

- Improving training data and feature selection: Properly manipulate and visualize data; Select features based on NLP methods for each model and feature importance to avoid the dimensionality and potentially overfitting issues.

- Model and result evaluation: Understand and implement advanced NLP models, and evaluation models based on their confusion matrix, ROC curves and precision-call curve (Based on course materials). And use GPU to improve training speed.

Data Preprocessing

Let’s first take a look at our data.

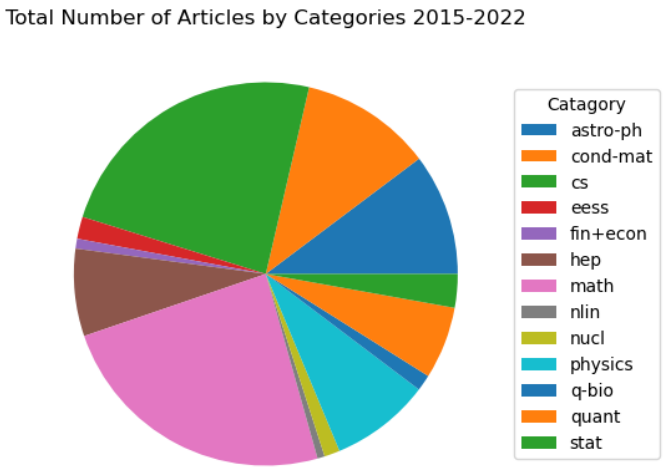


There are a total of 2,171,090 rows of data contained in the arxiv database from 1986 to 2022, the graphs above show the total and growth number of articles by categories since 2015 (1,169,787 data points). The number of articles grew each year across categories and reached 1,169,787 in 2022. The categories used for plotting graphs and training are shown below:

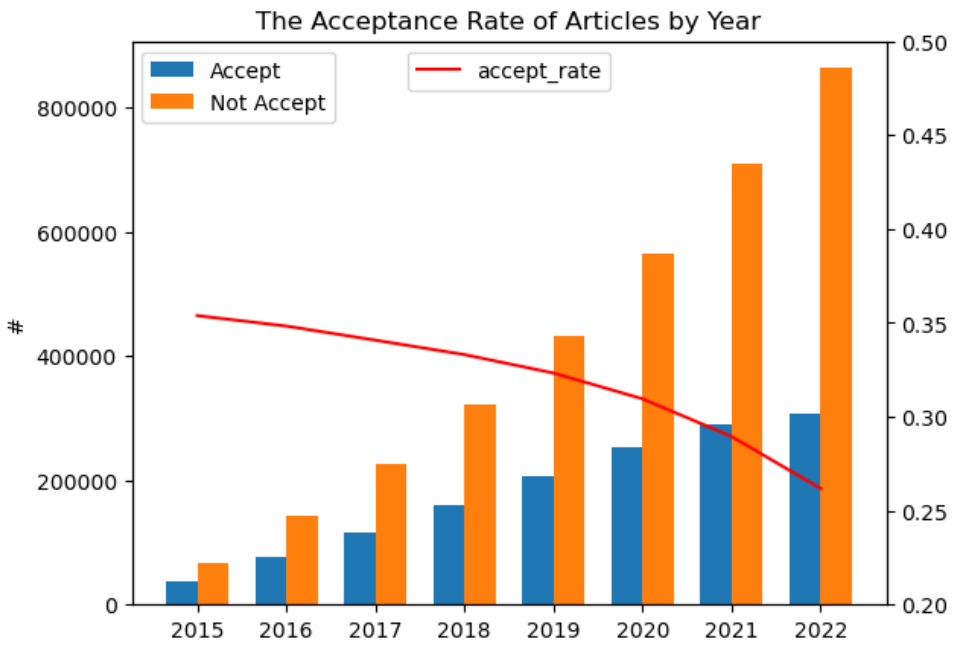
Categories	Categories in Dataset	Categories in Training
High Energy Physics	hep-ex, hep-lat, hep-ph, hep-th	hep
Mathematics	math-ph	math
Nuclear	nucl-th, nucl-ex	nucl
Quantitative Finance, Economics	q_fin, econ	fin+econ
General Relativity and Quantum Cosmology,Quantum Physic	gr-qc,quant-ph	quant
Condensed Matter	cond-mat	cond-mat
Quantitative Biology	q-bio	q-bio
General physics	physics	physics

Nonlinear Sciences	nlin	nlin
Astrophysics	astro-ph	astro-ph
Computer Science	cs	cs
Statistics	stat	stat
Electrical Engineering and Systems Science	eecs	eecs

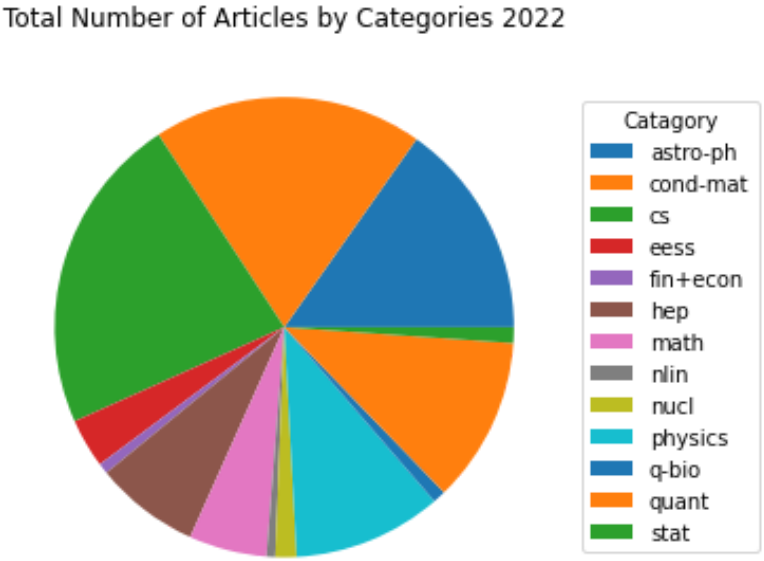
For getting precise and accurate predictions, the categories in the dataset were merged and transformed based on their similarity and reduced the categories from 38 to 13 defined as main categories for training.



Based on the main categories we defined, the pie graph above shows the proportion of articles in each category compared to the total number of articles from 2015 to 2022. The top 4 main categories are ”math”(24.10%), “cs” (23.19%), “cond-mat”(11.10%), and “astro-ph”(10.27%), while most of other categories’ proportions are lower than 10% and the lowest is for main categories “nlin” with only 0.61%.

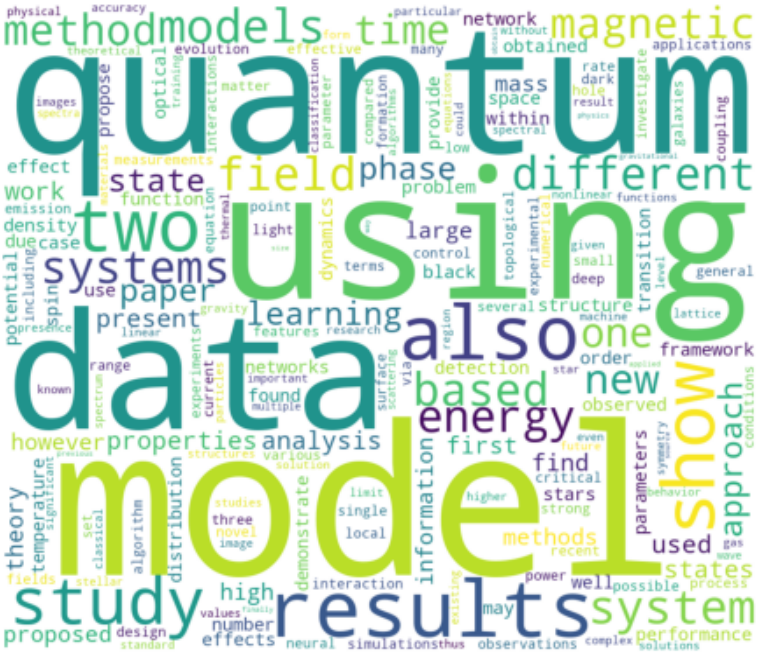


Then, given the growth of the number of articles over the years, we want to discover exactly how many papers were accepted and published in journals from 2015 to 2020. From the graph above, although the “accepted” articles increased from 37,204 to 306,249, the “not accepted” articles increased from 67,926 to 863,583, which caused the percentage of journal acceptance to decrease from 35.39% to 26.18%. With massive data in the dataset, we specifically wanted to know how the most recent (year 2022) data categories are distributed and predicted by the title and abstract of each article in 2022.



With total number of 17,205 articles in the arxiv database so far, The top 4 main categories are changed to ”cs” (21.80%), “cond-mat” (19.16%), “astro-ph”(15.05%), and “quant”(11.86%), while the lowest proportion in main categories is “nlin” with only 102 (0.60%) articles.

Before explaining and building the model, the titles and abstracts of the articles are combined into “text” column and preprocessed by removing all the special character, punctuation, spaces from string and stopwords (a list of commonly used words such as “a”, “an”, “in” that are often ignored in text-related analysis). In the final step, we summarized the “clean text” with the top 10 most common words with counts: ('model', 9769), ('quantum', 7934), ('using', 7840), ('data', 7720), ('results', 6428), ('also', 5928), ('two', 5585), ('show', 5457), ('study', 5161), ('system', 5115).
Visualize result with wordclouds:



Model Methodologies

Naive Bayes Model

Naive Bayes is a simple probabilistic classifier that is based on the application of Bayes' theorem with the assumption of independence between the features. It is a popular method for text classification, as well as many other types of classification tasks.

In text classification, a Naive Bayes model is trained on a dataset of labeled text examples and their corresponding labels. During training, the model estimates the probability of each word

occurring in each class, as well as the prior probability of each class. Given a new piece of text and a set of possible labels, the model can then use Bayes' theorem to compute the posterior probability of each class given the words in the text. The class with the highest posterior probability is then chosen as the predicted label for the text.

One of the advantages of Naive Bayes is that it is simple and easy to implement, making it a good choice for text classification tasks with small to medium-sized datasets. It is also relatively fast to train and predict, making it suitable for real-time applications. However, the assumption of independence between features can be a strong limitation, and the model may not perform as well as more sophisticated approaches on more complex datasets.

TF-IDF

Tf-idf (term frequency-inverse document frequency) is a statistical method to numerically reflect how important a word is to a document in a collection of corpus. This method is often used as a weighting factor in searches of information retrieval, text mining and user modeling. By increasing the number of times a word appears in the text and offsetting the number of text in the corpus that contain the word, it helps us to adjust for the fact that some words appear more frequently in general. In the project, the Tf-Idf vectorizer with a limit of 8000 words is used to capture the words frequency and features; After that, the Chi-Square test is performed to drop some columns and reduce the matrix dimensionality based on the independence.

Bert Pre-training Model

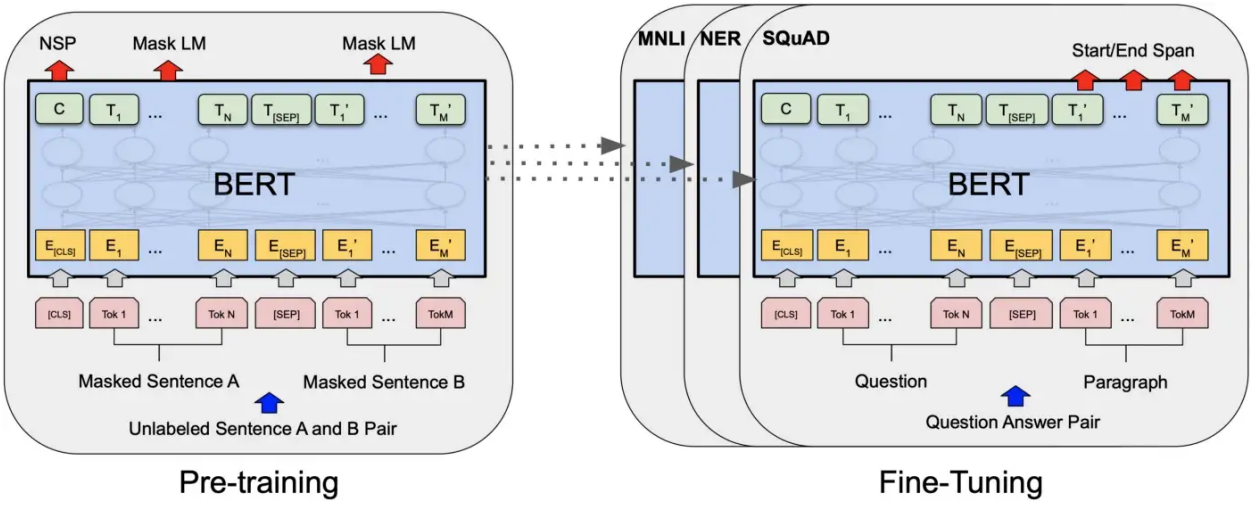


Image from [(Devlin et al., 2019)](#).

BERT is a language representation model that has been trained on a large dataset of text and has the ability to encode the context and meaning of words in a way that is useful for many natural language processing tasks. Text classification is one such task where BERT can be used.

To use BERT for text classification, you will first need to fine-tune the model on a labeled dataset of text examples and their corresponding labels. This involves adjusting the weights of the BERT model to optimize its performance for the specific task of text classification. There are a few different ways to fine-tune BERT for text classification, such as using a task-specific classification head on top of the BERT model, or fine-tuning the entire model end-to-end.

Once the BERT model has been fine-tuned for text classification, you can use it to make predictions on new, unseen text examples. Given a piece of text and a set of possible labels, the model will output a probability distribution over the labels, indicating the likelihood that the text belongs to each class. You can then choose the label with the highest probability as the predicted class for the input text. In this work, we use a variant of Bert called “Distilbert” to do all tasks

Huggingface Transformers

In this project, we use PyTorch based Hugging Face Transformers to do text prediction. Hugging Face Transformers is a library of pre-trained models and tools for natural language processing (NLP) tasks developed by Hugging Face, a startup company based in New York City.

The library is built on top of the popular deep learning framework PyTorch and includes a large collection of pre-trained transformer-based models that have been trained on a variety of NLP tasks. These models can be used for a variety of tasks, including language translation, text classification, summarization, and question answering.

In addition to the pre-trained models, the Hugging Face Transformers library also provides tools for fine-tuning the models on specific tasks, as well as tools for data preprocessing, evaluation, and visualization.

Model Evaluation

Results and Analysis

In this section, we show all the results and analysis of our models on different types of tasks.

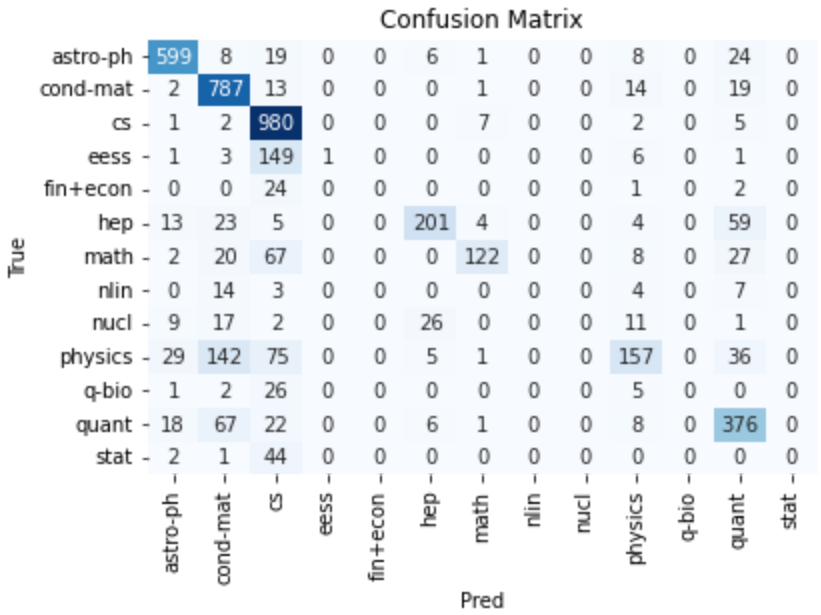
Paper Category Prediction

Models	Features	Number of Labels	Accuracy
Naive Bayes	TF-IDF of title and abstract	13	73.94%
Distillbert	TF-IDF of Title and abstract	13	76.1%
Distillbert	Title and abstract	13	77.5%

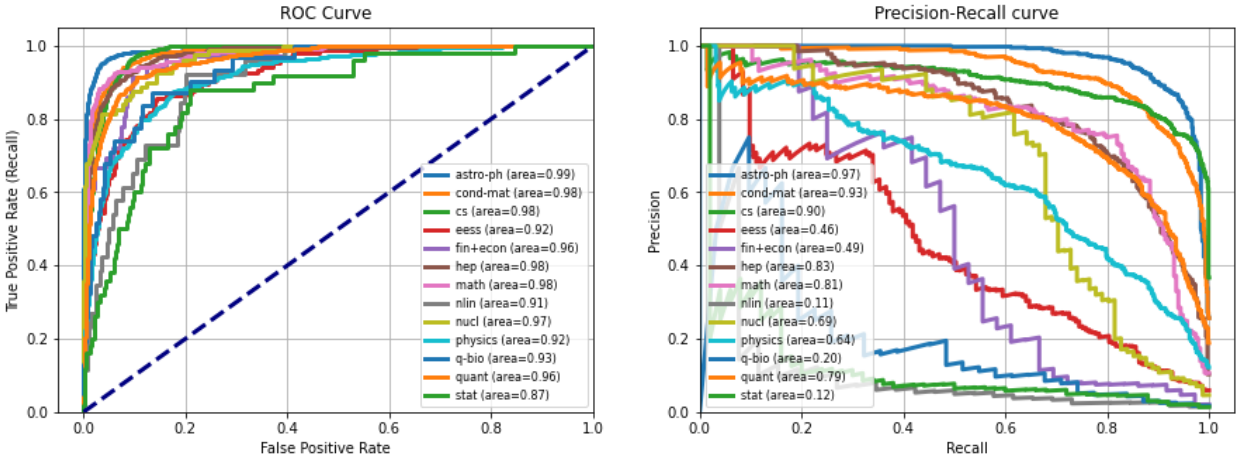
We compare naive bayes model with neural based distillbert model on paper category prediction based on paper's title and abstract. For Naive Bayes classification model, we use TF-IDF over the concatenation of title and abstract as features. For Distillbert, we directly use the concatenation of title and abstract as features.

From this table, we can see that the Distillbert based model performs better than Naive Bayes. That's because Distillbert is a transformer-based model, which means it can process input sequences of any length and attend to all tokens in the sequence simultaneously. But Naive Bayes can only take a subset of text sequences as input. Overall, BERT is generally considered to be a more powerful and effective tool for text classification than Naive Bayes, especially for tasks that require a deep understanding of the input data. So for the rest of the tasks, we conduct experiments only with Distillbert.

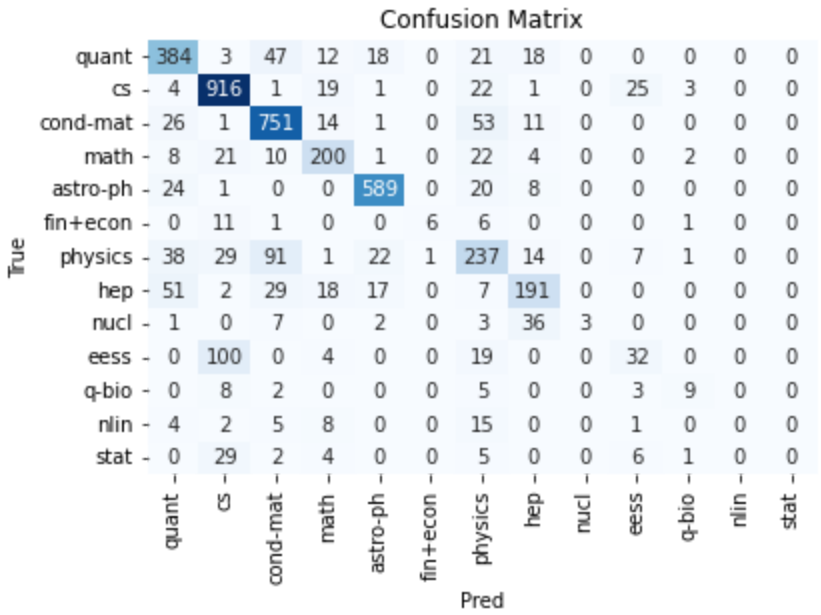
The following figure is the confusion matrix of the predictions of Naive Bayes model. We can see that the data is pretty unbalanced. There are much more papers in CS, QUANT, COND-MAT, ASTRO-PH categories than others. There are some categories that have no predictions, such as FIN+ECON, NLIN, NUCL, Q-BIO.

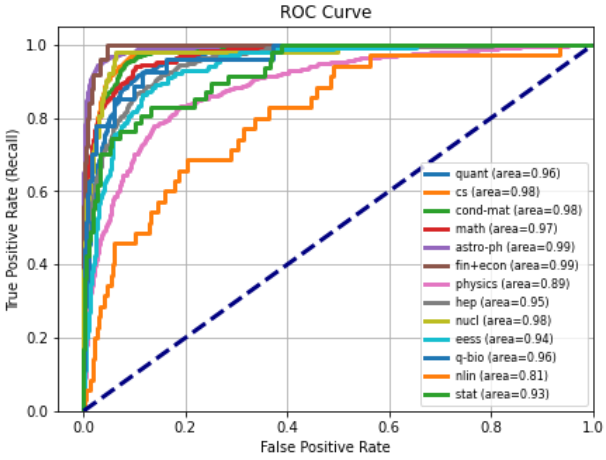


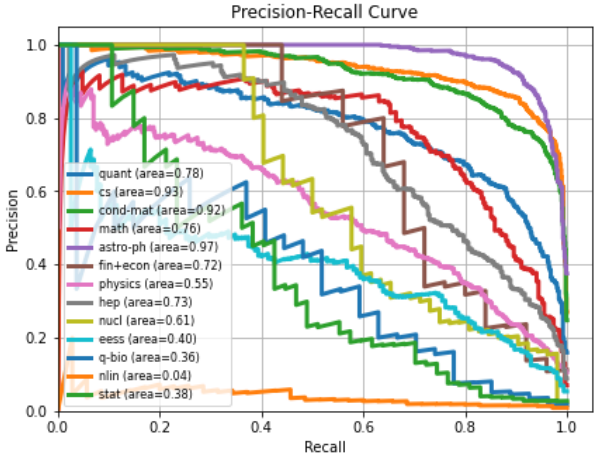
The following figure shows the ROC curve and PR curve. We can see that different categories have very different accuracies. That's because the training data is also not very balanced.

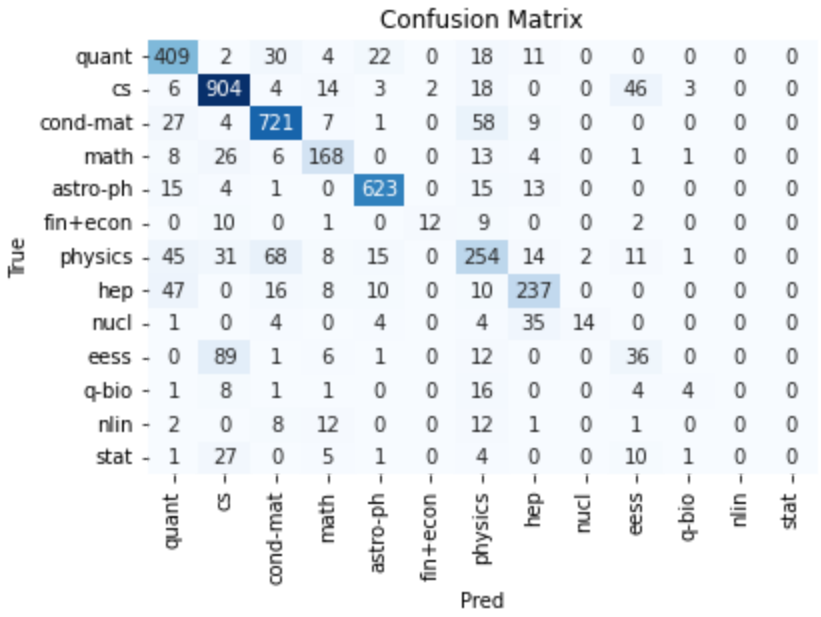


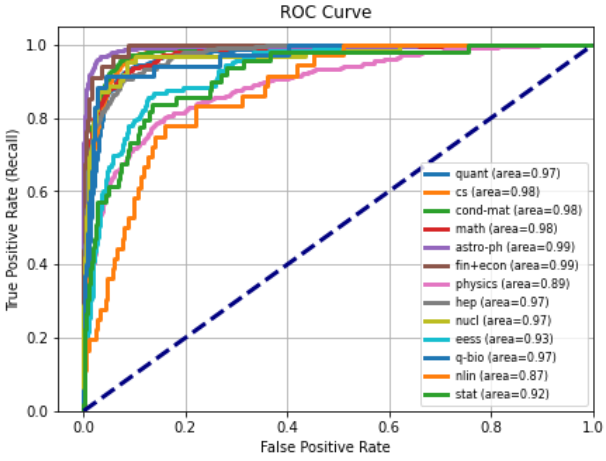
The following figures show a similar story with Naive Bayes model.

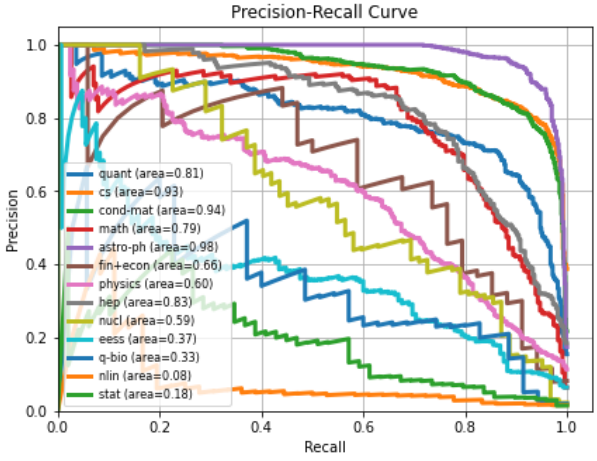

Confusion Matrix
quant 384 3 47 12 18 0 21 18 0 0 0 0 0
cs 4 916 1 19 1 0 22 1 0 25 3 0 0
cond-mat 26 1 751 14 1 0 53 11 0 0 0 0 0
math 8 21 10 200 1 0 22 4 0 0 2 0 0
astro-ph 24 1 0 0 589 0 20 8 0 0 0 0 0
fin+econ 0 11 1 0 0 6 6 0 0 0 1 0 0
physics 38 29 91 1 22 1 237 14 0 7 1 0 0
hep 51 2 29 18 17 0 7 191 0 0 0 0 0
nucl 1 0 7 0 2 0 3 36 3 0 0 0 0
eess 0 100 0 4 0 0 19 0 0 32 0 0 0
q-bio 0 8 2 0 0 0 5 0 0 3 9 0 0
nlin 4 2 5 8 0 0 15 0 0 1 0 0 0
stat 0 29 2 4 0 0 5 0 0 6 1 0 0
True
quant cs cond-mat math astro-ph fin+econ physics hep nucl eess q-bio nlin stat
Pred


ROC Curve
True Positive Rate (Recall)
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
quant (area=0.96)
cs (area=0.98)
cond-mat (area=0.98)
math (area=0.97)
astro-ph (area=0.99)
fin+econ (area=0.99)
physics (area=0.89)
hep (area=0.95)
nucl (area=0.98)
eess (area=0.94)
q-bio (area=0.96)
nlin (area=0.81)
stat (area=0.93)


Precision-Recall Curve
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0
quant (area=0.78)
cs (area=0.93)
cond-mat (area=0.92)
math (area=0.76)
astro-ph (area=0.97)
fin+econ (area=0.72)
physics (area=0.55)
hep (area=0.73)
nucl (area=0.61)
eess (area=0.40)
q-bio (area=0.36)
nlin (area=0.04)
stat (area=0.38)







Paper Acceptance Prediction

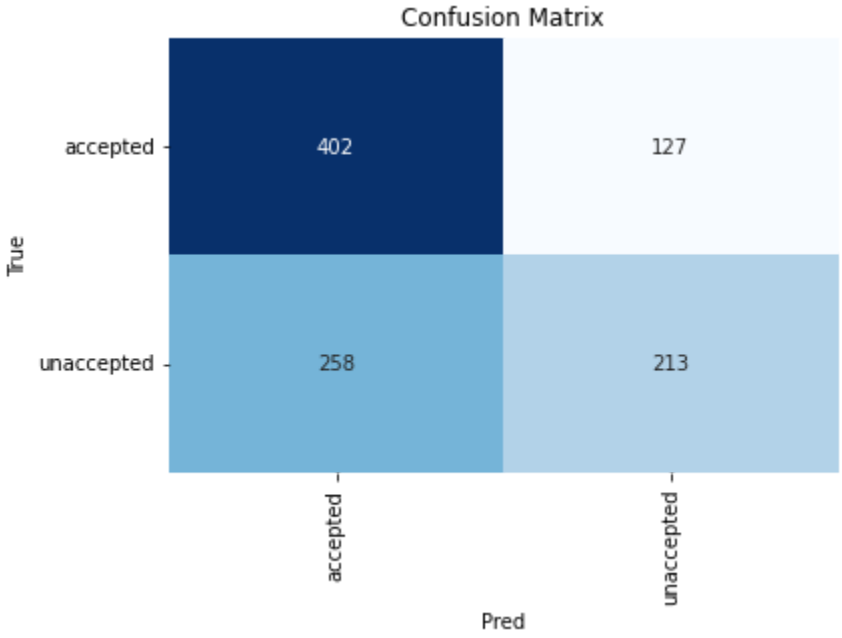
In this section, we try to predict whether these papers can be accepted. We labeled papers with journal information as accepted papers and papers without journal information as unaccepted papers.

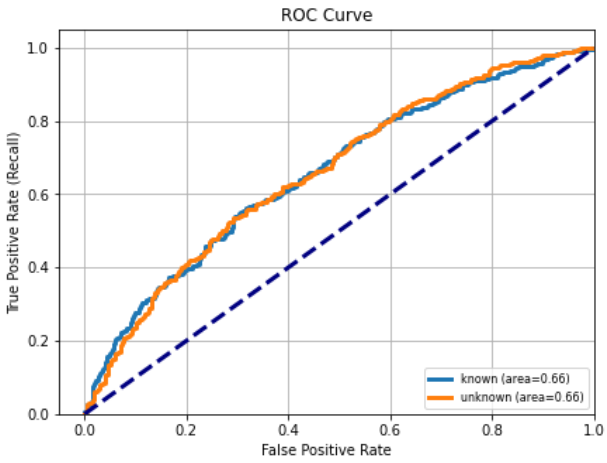
The following table shows the results of the paper acceptance task. Because of the limitation of computation resources, we only use 10,000 of data for these experiments. We can see that when we use abstract features, the prediction accuracy is higher than just using title as input. This is unstandable because abstracts contain much more information than titles.

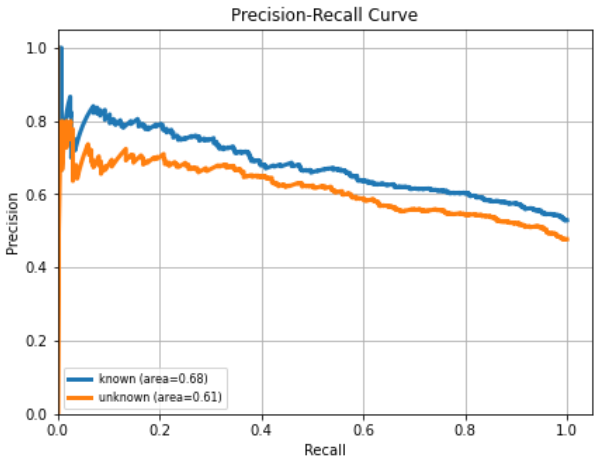
Model	Total	Feature	Labels	Accuracy	Training	Test-se

	data size				set size	t size
Distillbert	10k	Title	accepted, unaccepted	61.5%	9000	1000
Distillbert	10k	Abstract	accepted, unaccepted	66.1%	9000	1000

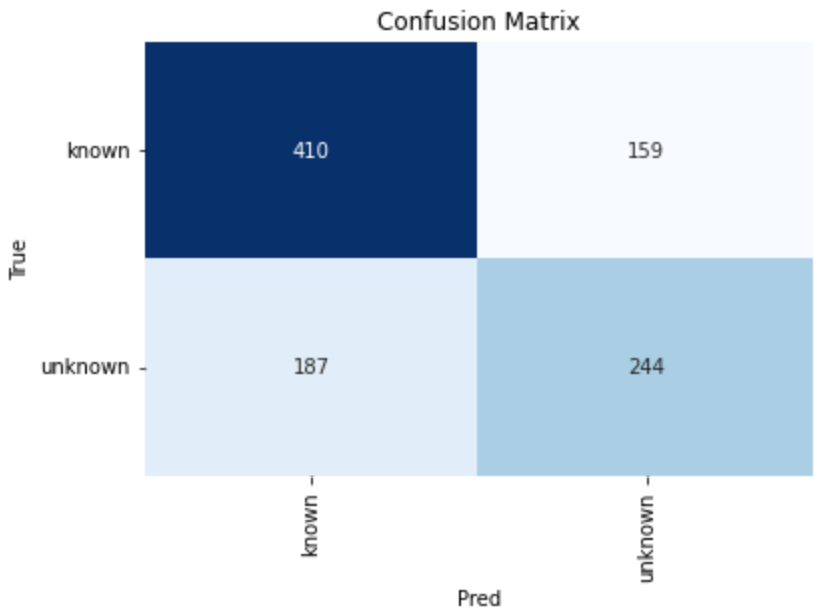
The following confusion matrix shows the confusion matrix, ROC curve and PR-curve for the experiment with title as input.

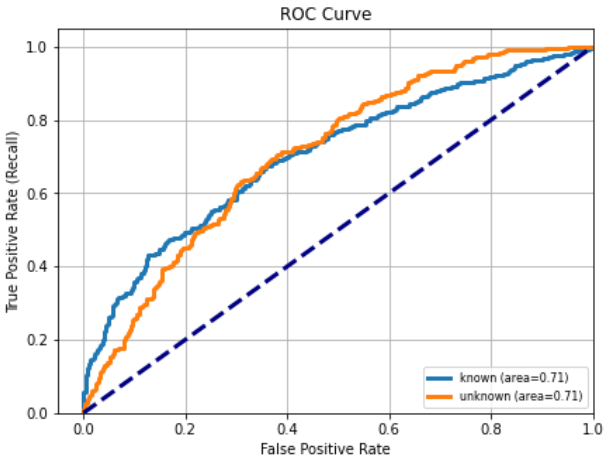


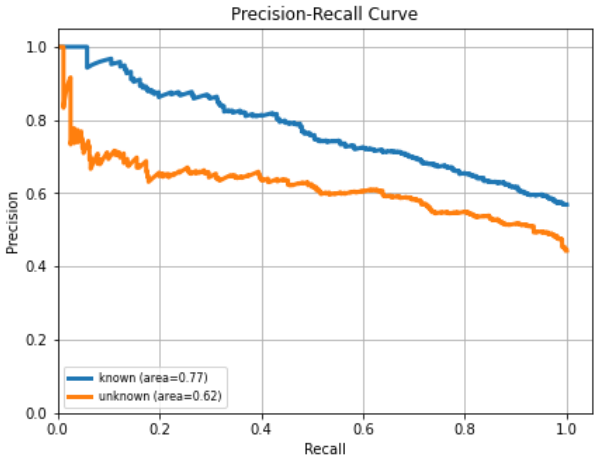




The following confusion matrix shows the confusion matrix, ROC curve and PR-curve for the experiment with abstract as input. We can see that the performance is better than the previous one.





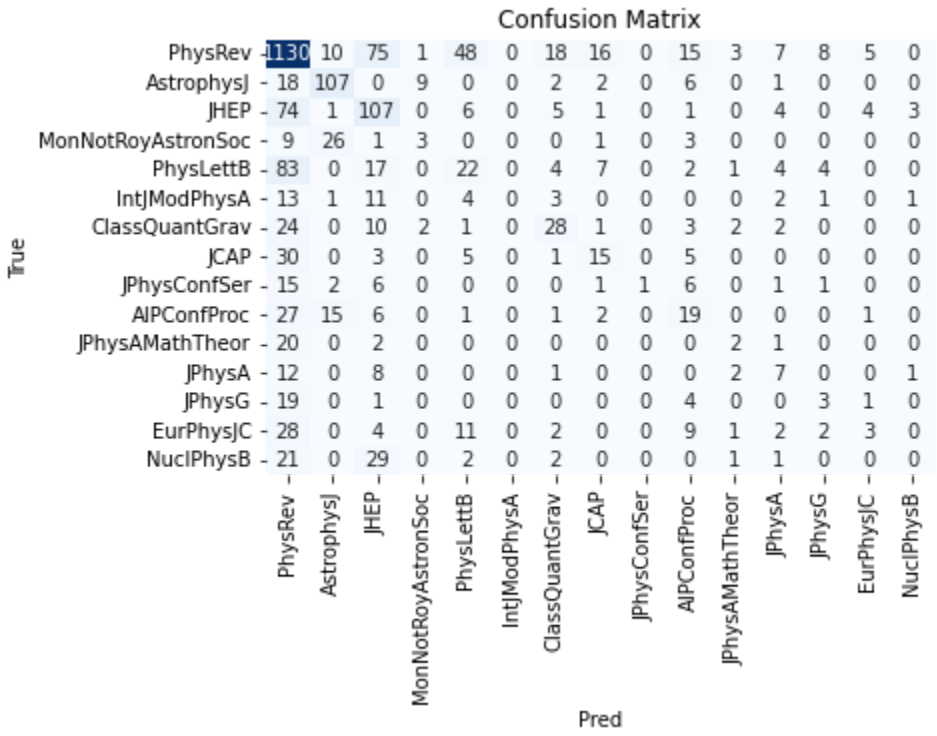


Accepted Journal Prediction

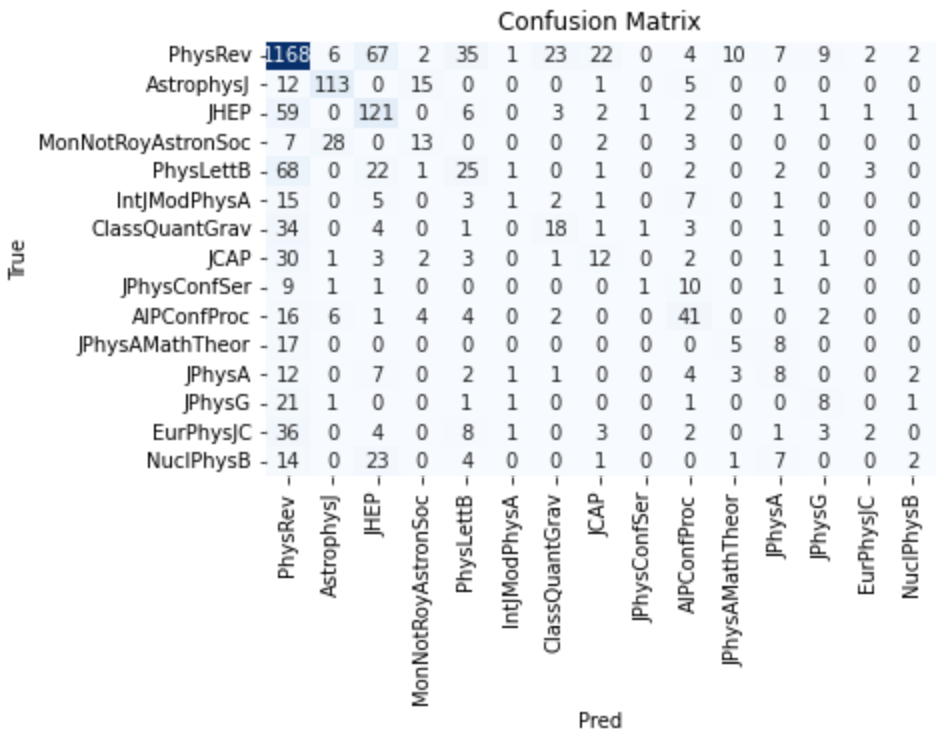
In this section, we introduce the results of journal prediction.We conduct experiments on all accepted papers. We pick the top 2 and 15 journals in the arXiv dataset as training data. The following table shows the results. From the results, we can see that predicting the corresponding accepted journal from 2 candidates is a very easy task. The model can easily make the right prediction with 95.3% accuracy with only 1520 training samples. However, when we need to predict the right journal from 15 candidates. The accuracy drops quickly. With 2523 training samples, the accuracy is 59.7%. When we add more training samples to 21138. The accuracy achieves 64.2%. If we further use abstract features. The accuracy can be boosted to 66.7%.

Feature	Number of labels	Accuracy	Training set size
Title	2	95.3%	1520
	15	59.7%	2523
	15	64.2%	21138
Abstract	15	66.7%	21138

The following figure shows the confusion matrix of models trained with 15 labels. We can see that PhysRev is very dominant in this data. But our model can still correctly predict those journals with much less data.



The following figure shows the results of using abstract features. We can see more true positive cases in this figure.



Potential Next Step

What we have done is just a simple initial step. As potentials next steps, we can do the following directions:

1. Try to do larger scale experiments involving more journals as labels and also more training samples. From our experiments, we show that training with more data, the model can achieve higher accuracy. It will be interesting to involve even more data and see whether it is useful for real world applications.

2. Design a label system with a hierarchical structure. For example, Computer Science -> Artificial Intelligence -> Natural Language Processing -> NLP Journal Names. This will be very helpful for researchers and maybe can also further improve the model performance.

3. Use paper content instead of title and abstract to see the accuracy. Paper content is much more important than title and abstract for a paper. It will be much more reasonable to use paper content as features.

4. Balance the sample data before training the models. Since there are different numbers of articles in different categories, and the acceptance rates also vary over years, to effectively oversampling or undersampling to balance the training data or to adapt the model that incorporates a penalty to compensate for the fact that there is an imbalance might be essential and improve the results. Besides that, the overall trends of the

numbers of articles in different categories over years and the change of acceptance rates over years (time-series and some seasonality factors) may need to be captured to improve the prediction results.