Exploring Machine Learning Techniques to Improve arXiv Paper Categorization

Junwon Choi

Math 156: Machine Learning
Department of Mathematics
University of California, Los Angeles

December 3, 2023

Exploring Machine Learning Techniques to Improve arXiv Paper Categorization

1 Abstract

Hundreds of research papers and scholarly articles are uploaded each day across countless journals. With such a vast amount of available texts, it can pose a problem for researchers hoping to use the available information. How can one be expected to efficiently navigate such a huge number of articles? In this paper we propose a multi-step solution to this problem within the ArXiv paper repository. Firstly, we explore the effectiveness of 2 unsupervised clustering methods (k-means, DBSCAN) to determine if a more simplistic categorization system can provide a similar differentiation of papers. Second, we integrate the ANNOY (Approximate Nearest Neighbors) model to introduce a tool which allows for the fast recovery of papers most similar to one's own to aid in research that may span several distinct categories. Finally, we use 3 supervised learning methods (KNN, random forest, fastText) to create a tool to assist researchers in self classifying their own research papers so as to be most easily accessible to others.

2 Introduction

The ArXiv repository was founded in 1990 by a researcher at the Los Alamos National lab hoping to develop a way to more efficiently send preprints of physics papers to his colleagues. What began as an email tool built for convenience has since grown into one of the largest and most commonly used sources to access academic papers and is now owned and operated by Cornell University. Hosting well over 2.4 million scholarly articles in the, the ArXiv system spans a huge number of years and scientific research areas, providing an invaluable resource for researchers finding references through the literature of their field [2].

While the increase in availability of research papers and the growth of this repository has been an undeniable benefit for scientific research, the huge expansion of the archival system has introduced some issues. A system that initially held a small number of niche physics papers now maintains papers in a far more broad range of subjects, including mathematics, computer science, quantitative biology, quantitative finance, statistics, electrical engineering and systems science, and economics [2]. Within each of these research areas, there exists sometimes 50 or more labels to further subset the papers. Currently, papers can be classified into any of over 150 distinct categories, which are to be defined by those who upload each paper [1].

This large number of distinct categories poses several difficulties for researchers. How can one be expected to effectively navigate such a large number of classes when performing research? Researching for a paper which may span several categories can be a daunting task. One must search through each of many potential categories, for papers which are similar to their area of interest. Furthermore, how does one know which categories to assign to their own papers? Of course a researcher knows the topic of their own research, but how might they go about classifying their papers into potentially several subject areas out of several hundred? To be able to make ones papers more easily accessible to those whom it may benefit, the process of self classifying may not span all important areas. For example a researcher may classify their machine learning paper as a mathematics paper, limiting access to those who may be searching under the computer science section of ArXiv. This leaves room for human error that will require retroactive adjustments. Lastly, is there an efficient way to organize such a great variety of papers, while reducing the number of difficult to navigate categories? In this paper, we explore a number of solutions to each of these problems, employing unique machine learning methods at each step.

Firstly, to address the problem of discovering research papers, we introduce a novel tool leveraging the ANNOY (Approximate Nearest Neighbors) model stacked with a clustering model (k-means, DBSCAN) to effectively and efficiently pull relevant research papers from the dataset. This tool's key

feature is the implementation of the computationally inexpensive ANNOY model as well as Term Frequency - Inverse Document Frequency (TF-IDF) representations of research paper abstracts to quickly retrieve a specified number of mostly similar research papers (similar content and categorization) based on a given input abstract. The goal of this tool is to provide a fast and simple way to query the ArXiv database, ideally saving significant time that would otherwise be spent searching for papers.

To solve the issue of self assignment, we propose a second tool. Utilizing several supervised learning models, we create a system which allows users to provide the abstract of their own papers and receive an estimation of n most relevant categories, where n is a parameter that can be specified by the user. One can utilize this tool to best categorize their research publication. Initial attempts at this involved the use of random forest and K-Nearest Neighbor models for multiclass classification. However, best results were achieved by employing fastText model, which allows for efficient text classification into multiple classes, and also provides a probabilistic representation of the research categories.

To determine the most suitable model for such tasks, we tested multiple models against one another after also running parameter tuning tests for optimal model settings, as well as dimensionality reduction of training datasets. The results of this analysis provides valuable information to the ArXiv system for simplifying their repository, as well as providing an analytical framework that can be applied to similar multiclass datasets looking to simplify their organization.

3 Background

Text classification and Natural Language Processing have transformed how we organize and understand large amounts of textual data, particularly with the goal of improving access to scientific literature. Researchers have developed and refined various machine learning models and techniques to apply text classification to automatically categorize scholarly articles into specific subject domains. One such basic algorithm is the classical decision tree. But more notably, popular models include k-nearest neighbors (KNN), Naive Bayes, and Support Vector Machines (SVMs), which are supervised methods that learn the patterns and relationships within text, making them able to discern the distinctions across diverse subject categories [4].

"Article Classification using Natural Language Processing and Machine Learning" by Thanh et al. compares the performance of these three algorithms in particular and concludes that SVM produces the best classification results, but is also dependent on the data pre-processing and should be explored with larger and more diverse datasets [8]. The data pre-processing, which we referenced in our models as well, involves the segmentation and vectorization of the raw text data. Although not implemented into our pipeline, it is also important to note that when using the SVM classifier, using singular value decomposition to analyze and shorten the dimension of the characteristic space helps to improve effectiveness, which is why we tested models using datasets with reduced dimensions our selected models. In summary, the combination of natural language processing and machine learning algorithms is proven to be effective for developing automated classification systems for scholarly articles.

Another paper that outlines what has been previously studied about article classification is "SemEval-2018 Task 7: Semantic Relation Extraction and Classification in Scientific Papers" by Gabor et al., which specifically relates to the task of identifying and classifying semantic relations between concepts found in abstracts into discrete categories [3]. To train the model, first, pairs of relation instances were manually annotated and classified based on semantic relation typology as defined by a table in the paper [3]. For the testing data, only annotation was completed and the classification was left incomplete [3]. The authors then repeated this subtask with a set of "noisy" data, where the occurrences were automatically annotated, but the semantic relations were still done manually for the training data [3]. The results of this study were relevant to our project in that it provided more information about the nuanced information

that can be extracted from the text dataset we were using. While the goal was slightly different from ours, the information learned helped us to develop a stronger understanding of how we could manipulate and process our data for our tasks.

4 Dataset

4.1 Dataset Description

This section will describe the ArXiv dataset, which is the dataset used for this project. The dataset is publically available and can be accessed through either the ArXiv API or Kaggle. The dataset available on Kaggle contains 1.7 million articles, with each row consisting of information such as the title, author, categories, abstract, and full text PDF. There are 158 subcategories in total in the system [1]. For this project, we pulled articles that were published between 2018 and 2020 (inclusive), which subsetted 3 years of data with 15,771 articles. The data subsetting can be increased to account for more years if computation time is not a concern.

We set 3 different datasets for our models: the full dataset [B.1], an expanded dataset [B.2] with each article having distinct categories, and a main category dataset [B.3] with each article having only their main category. The full dataset was used for the unsupervised clustering models, while the expanded dataset and main category dataset were used for the supervised learning models. The previews for the three datasets can by found in appendix [B].

In the full dataset, the `category` column values are lists of all the categories that a particular article is in. For example, the first row in the full dataset has the value `[math.DG, math.AP, math.MG]` for the `category` column, which means that the article is in the `math.DG`, `math.AP`, and `math.MG` categories. The expanded dataset has the same columns as the full dataset, but the `category` column values are not lists, but rather a single category. The main category dataset has the same columns as the full dataset, but the `category` column values are not lists, but rather the main category of the article. For example, the first row in the main category dataset has the value `math` for the `category` column, which means that the article is in the `math` category. There are 3764 unique category combinations in the full dataset, 158 unique categories in the expanded dataset, and 20 unique categories in the main category dataset. We note that categories with less than 10 articles were removed from the dataset.

4.2 Dataset Preprocessing

The abstracts of each paper were processed to remove punctuation and stopwords and were set to lowercase. Vector representations of each preprocessed abstract are created using TF-IDF vectorization, a process built on the Bag of Words representation of text, in which strings of words are vectorized according to the number of times a word appears in a given document and the number of documents which contain this word. These vectorized representations are used for nearly all of the models that we implemented, with the exception of fastText, which uses a different vector in the form of word embeddings.

5 Methods

5.1 Category Reduction

Category reduction involves the application of various unsupervised clusterting methods to determine whether a more simple organization schema exists for the ArXiv repository. The goal of this step is to reduce the number of categories (category combinations) from currently 3764 to a more manageable number. This will allow for easier navigation of the repository, as well as a more efficient way to organize new papers. Here we explore the effectiveness of 2 unsupervised clustering methods: k-means, DBSCAN. To determine the effectiveness of both techniques, we use the silhouette score, which is a

measure of how similar an object is to its own cluster compared to other clusters, mimicking the analysis performed by Shahapure et al. [7]. Using the mean intra-cluster distance a and the mean nearest-cluster distance b for each sample, the score is calculated between each cluster, and the silhouette coefficient of a clustering assignment is the mean score between all clusters. The silhouette score between two clusters then is given by

$$s = \frac{b - a}{\max(a, b)}.$$

We evaluate each of the posed models with respect to their silhouette scores, and present the best results from each model.

The initial clustering model employed is k-means, an iterative process where the number of means is defined. In each iteration, a cluster's center is determined as the centroid of all points assigned to that cluster. Utilizing the k-means++ algorithm, an improvement over the original k-means, ensures better convergence and results. The algorithm initializes cluster centers by selecting the first one uniformly at random from data points and subsequent centers based on the distance from the closest existing cluster center. To find the optimal number of clusters, the elbow method is employed. This method calculates inertia (sum of squared distances of samples to their closest cluster center) for various cluster sizes. The optimal number of clusters is identified at the elbow point, where inertia begins to decrease more slowly. While a distinct elbow wasn't evident, 200 clusters were chosen, representing a point where inertia decreases relatively slowly. The inertia vs. clusters plot is available in appendix [C.1.1].

Additionally, we checked silhouette scores for different numbers of clusters, and found that the silhouette score was actually highest for 500 clusters. This is likely due to the fact that there are a large number of categories, and so the optimal number of clusters is likely to be much higher than the number of categories. We note that the trend on the plot is indicative of the model favoring a higher number of clusters than 500, but we chose to use 500 clusters for computational efficiency. This may be a point of improvement for future work. The plot of the silhouette scores against the number of clusters can be found in appendix [C.1.4] and the table of silhouette scores can be found in appendix [C.1.5]. The silhouette scores for 500 clusters is 0.0095.

We then tested the DBSCAN model. Before proceding, we reduced the dimensionality of the dataset using truncated singular value decomposition (SVD). This is a technique which reduces the dimensionality of the dataset by projecting it onto a lower dimensional space. This is done by computing the singular value decomposition of the dataset, and then keeping only the first k singular values. With the full dataset vectorized, there were 78504 features, which would be computationally expensive to run the DBSCAN model on. In appendix [C.3], we plot the explained variance against the number of components. Similar to the elbow method, we determine the optimal number of components at the point where the explained variance begins to decrease more slowly. We observed diminishing returns beyond 25 components, and also the cumulative explained variance plot that after 25 components displays linear increase in cumulative explained variance. Hence we reduced the dimensionality of the dataset to 25 components.

After applying the truncated SVD, we tested models with ranging epsilon and minimum sample values to find the optimal parameters for the DBSCAN model. The DBSCAN model is a density-based clustering algorithm, which groups together points that are closely packed together, and marks points that lie alone in low-density regions as outliers. The model requires two parameters: epsilon and minimum samples. Epsilon is the maximum distance between two samples for them to be considered as in the same neighborhood. Minimum samples is the minimum number of samples in a neighborhood for a point to be considered as a core point. We tested the model with epsilon values ranging from 0.1 to 2.0 and minimum sample values ranging from 2 to 8. The best silhouette score was 0.3463 with epsilon = 0.2 and minimum samples = 2. There are a number of reasons why we felt that DBSCAN may out-

perform other clustering algorithms. Firstly, this method does not require us to specify the number of clusters, but rather the number is determined automatically as a result of the hyperparameters. In this application, it may be helpful to have the number of clusters determined by the model, as we are hoping to achieve an optimal number of clusters. Additionally, DBSCAN can learn arbitrary cluster shapes, which may be useful to this application. Further, it is robust to outliers, which may be beneficial if the paper labelling is not standardized.

Although the DBSCAN model had the highest silhouette score, we chose to use the k-means model for the paper retrieval tool. This is because the k-means model was more robust in that it kept more possible categories while DBSCAN set an extensive amount of categories to be outliers/noise. This is likely due to the fact that the k-means model is more sensitive to the number of clusters, and so we chose to use the k-means model with 500 clusters.

5.2 Paper Retrieval Tool

Rather than using a traditional nearest neighbors calculation, our paper retrieval tool leverages the approximate nearest neighbor model (ANNOY) to quickly retrieve a specified number of mostly similar research papers (similar content and categorization) based on a given input abstract. The ANNOY model is a computationally inexpensive model that is able to quickly retrieve the most similar papers to a given input abstract. The model works by creating a tree of binary splits of the data, and then traversing the tree to find the nearest neighbors. The model is able to find the nearest neighbors in $O(\log N)$ time, where N is the number of data points. This is significantly faster than the traditional nearest neighbors calculation, which has a time complexity of $O(N)$.

5.3 Topic Classifer

The purpose of the topic classifier is to provide researchers with the option to quickly and easily obtain a list of potential paper classifications. This allows those who upload to repositories like ArXiv to ensure that they assign the best possible list of subject classifications to their research. The goal of this tool is to provide researchers with a list of likely subject classifications, some of which may not have originally been considered. At this stage, we applied models to the expanded and main category datasets. The performance was evaluated using the precision metric, which is the ratio of true positives to the sum of true positives and false positives.

Our first attempt involved a K nearest neighbors (KNN) model, using the vectorized abstracts to determine the classification of a new datapoint by the most common classification of its nearest neighbors. The KNN model determines the euclidean distance between the new example and all other datapoints, focusing on the K nearest neighbors, where K is a specified parameter. The classification of the new datapoint is a product of the classifications of the neighbors. The best result was found when using the 15 nearest neighbors. This model, while relatively simple, still achieves rather accurate results due to the effectiveness of the vectorization.

A second attempt involved the usage of a random forest classifier. The random forest classifier involves the training of many different decision trees. Each tree is trained on a random subset of the data, and the final classification is determined by the majority vote of all the trees. This model is effective at reducing overfitting, and is also relatively fast to train.

The final implementation of this tool employs the fastText model by Joulin et al. from Facebook AI Research Lab [5]. This model addresses baseline drawbacks through hierarchical softmax and n-gram features. To enhance generalization, the linear classifier is factorized into low-rank matrices, a method adopted by the fastText model. It uses matrix A as a word look-up table, averaging word representations into a text representation. This representation, similar to the Continuous Bag of Words (CBOW) model,

is fed into a linear classifier [5]. The model utilizes the softmax function f to compute the probability distribution over predefined classes, minimizing negative log-likelihood over N documents:

$$-\frac{1}{N}\sum_{n=1}^{N} y_n \log(f(BAx_n)).$$

For efficiency, when the class count is large, the fastText model employs hierarchical softmax, reducing computational complexity to $O(h\log_2(k))$. This hierarchical softmax utilizes a binary tree, providing advantages in both training and test times. The second component, n-gram features, addresses the limitation of the bag of words model in capturing word order. The fastText model incorporates n-grams to capture local word order, utilizing a hashing trick for efficiency [6]. This approach proved effective in terms of training time and results. This is important because ArXiv has a massive vocabulary with niche words and subject-specific vocabularies. This model proved to be the most effective, both in training time and results.

6 Results

6.1 Category Reduction

The results of the unsupervised portion provided some information regarding the feasibility of reducing the number of subjects in the dataset, and also provided an effective labelling schema to use with the ANNOY model. The k-means model was able to reduce the number of categories from 3764 to 500, which is a significant reduction. The DBSCAN model was able to reduce the number of categories from 3764 to 2, which is a very significant reduction. However, the DBSCAN model set a large number of categories to be outliers/noise, which is not ideal. In our most successful model, over half of the datapoints were considered to be noise. We concluded that the structure of the vectorized texts is not appropriate for applying this model, and instead we continued with K-means. Throughout testing, we applied the following article abstracts as sampling inputs. Note that all three articles are out of the scope of our training data, which in theory should represent a totally new research abstract to the model.

1. "Pretest and Stein-Type Estimations in Quantile Regression Model" - 2017 / Statistics Theory (math.ST)

2. "Smoothed GMM for quantile models" - 2017 / Statistics Theory (math.ST); Econometrics (econ.EM); Methodology (stat.ME)

3. "Testing High-dimensional Covariance Matrices under the Elliptical Distribution and Beyond" - 2017 / Statistics Theory (math.ST)

The k-means model classified the first two samples into the same cluster, while the DBSCAN model classified all three samples as outliers/noise.

6.2 Paper Retrieval Tool

The paper retrieval tool is an application of the ANNOY model trained on our best performing k-means model. Its purpose is to present similar papers to a researcher to allow for faster research, and potentially more broad or multidisciplinary papers that would otherwise not be explored. Because of the nature of this tool, it is difficult to apply a quantifiable measure of its success. However, in initial testing, we found that the model proved effective in retrieving relevant papers. The model often pulls from papers within the same niche category, though the tool has also shown successes in returning papers classified into other subjects which still prove relevant to the inputted texts.

In appendix [D.2] we show an example of the usage of this system by inputting the first sample abstract for the original fastText paper. Note both the titles and categories of the suggested papers. We

observe that the relevant papers are subjects which are relevant to the fastText paper. As expected, there are suggested papers in math.ST, however we also note the recommendation of papers in subjects that we may not have otherwise considered, including econ.EM and stat.CO. We were impressed with the results of this model. It consistently and quickly provides a researcher with similar, relevant papers, and has potential to lead a researcher to explore new paths and take a more interdisciplinary approach.

6.3 Topic Classifer

The supervised learning portion of our paper involved the application of 3 distinct models on the specific and broad categories of the ArXiv papers. Each model used an 80-20 train test split, and accuracy of each model was measured as a proportion of the test set correctly classified. For reference, there are 157 specific categories, and 21 broad categories. Our first attempt used a random forest classifier. This model showed very poor performance in both metrics. The classifier achieved only 11.525% accuracy on the specific categories, and 50.825% accuracy on the broad categories. This model performed the worst of any of our models in both sections. Because random forest is a rather computationally expensive model, we were not able to train an extremely complex model, which may explain the results.

The second approach used a K nearest neighbor model in order to classify new abstracts. This model achieved far better results in the specific categories, achieving 20.342% accuracy, nearly double that of the random forest classifier. This model excelled in the broad classification, achieving 56.445% accuracy, which is the best performance in this section for any of our models. We believe that this performance is because the vectorized abstracts of similar papers would be largely similar within the broad categories. This is a result of many of these papers using the same vocabulary, with different broad categories having significantly different vocabularies.

Our third iteration used the fastText model. We used more extensive parameter tuning in this model, and were able to achieve the most notable results, with rather fast training time. Using 1 word n-gram and 40 epochs, we were able to achieve 27.179% accuracy on the specific categories. Using 1 word n-gram with 10 epochs we achieved 54.892% accuracy on the broad categories. We found that additional training epochs overfit the data and reduced performance on the test set.. While these results are far from perfect, the fastText model produces a probability distribution of the most likely categories. We can then use this distribution to to place papers into multiple categories, as is done in the current organization scheme. This makes the tool far more useful and applicable, as it allows for fast multiclass classification of new papers.

7 Conclusion and Future Work

In this paper we have explored numerous methods of improving the categorization of papers in the ArXiv repository. The paper retrieval tool allows users to more quickly find potentially relevant papers to their own, and it can be used to speed up the process of background research. The category assignment tool can be applied to new papers as they are uploaded to the dataset in order to assign each upload a number of different subjects. This tool can be used to make papers more widely available, as they may appear in subject queries that they otherwise would not. Future research in this area could involve the introduction of additional tools to manage such a dataset, for example an abstract generating tool, using NLP to generate concise and descriptive abstracts when input the content of a paper. Additional research could also be done in applying these processes to other large paper repositories.

8 Data Availability

ArXiv is an open-access archive for scholarly articles and the arXiv dataset is a freely-available repository of 1.7 million articles, containing paper features such as article titles, authors, categories, abstracts, and full text PDFs.

9 Contributions Statement

Each member of the group contributed roughly evenly throughout the course of the project. Initially we collaborated on background research and developing our questions. Once we began the project, we divided the coding work. Initial preprocessing and text vectorization steps were done by Mabel. She also aided in model research and selection for the next steps and led the development of the K-means clustering section. Mabel also wrote the background section of the report. Junwon implemented the ANNOY model as well as the FastText model, and led the overall organization of the code. This involved compiling all different workspaces into one cohesive document, and creating some missing visualizations. Reuben developed the DBSCAN model in the unsupervised portion. He also created the random forest and K nearest neighbors classifiers. Reuben also wrote the introduction to the paper. The work was split evenly, with members helping each other as was needed.

References

[1] arxiv api user's manual - arxiv info.

[2] arxiv.org e-print archive.

[3] Kata Gábor, Davide Buscaldi, Anne-Kathrin Schumann, Behrang QasemiZadeh, Haïfa Zargayouna, and Thierry Charnois. SemEval-2018 task 7: Semantic relation extraction and classification in scientific papers. In Marianna Apidianaki, Saif M. Mohammad, Jonathan May, Ekaterina Shutova, Steven Bethard, and Marine Carpuat, editors, *Proceedings of the 12th International Workshop on Semantic Evaluation*, pages 679–688, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[4] Emmanouil Ikonomakis, Sotiris Kotsiantis, and V. Tampakas. Text classification using machine learning techniques. *WSEAS transactions on computers*, 4:966–974, 08 2005.

[5] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification, 2016.

[6] Tomas Mikolov, Anoop Deoras, Daniel Povey, Lukas Burget, and Jan Honza Cernocky. Strategies for training large scale neural network language models. *2011 IEEE Workshop on Automatic Speech Recognition & Understanding*, pages 196–201, 2011.

[7] Ketan Rajshekhar Shahapure and Charles Nicholas. Cluster quality analysis using silhouette score. In *2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 747–748, 2020.

[8] Dien Tran Thanh, Bui Loc, and Nguyen Thai-Nghe. Article classification using natural language processing and machine learning. pages 78–84, 11 2019.

Appendix

A Code Pipeline

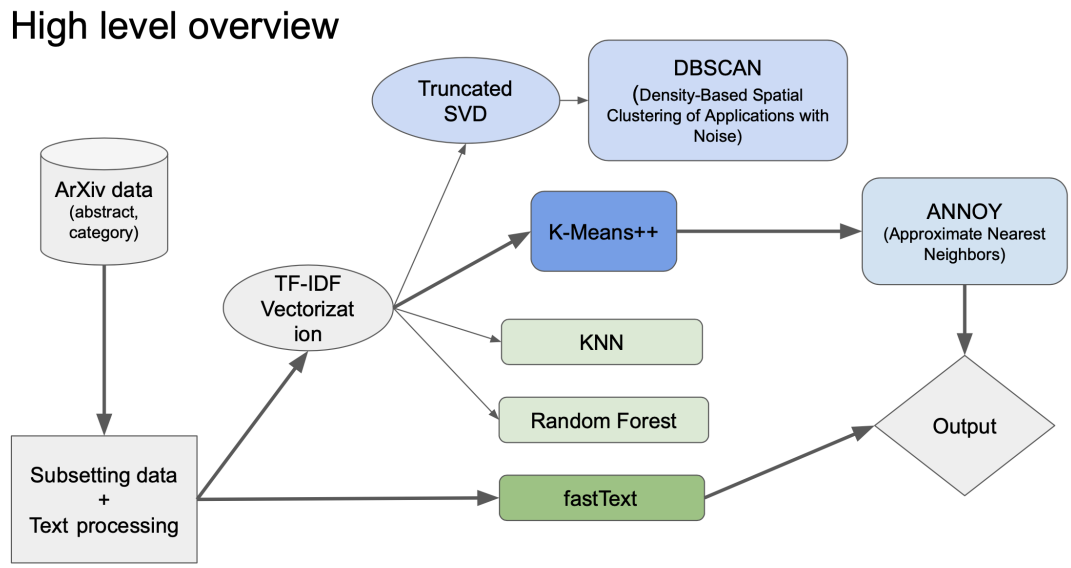

Figure 1: Code Pipeline - ArXiv Paper Categorization

B Dataset Preview

B.1 Full Dataset

id	title	abstract	category	year
0710.1849	Regularity of solutions of the isoperimetric p...	In this work we consider a question in the c...	[math.DG, math.AP, math.MG]	2018
0804.3104	Teichmüller Structures and Dual Geometric Gi...	The Gibbs measure theory for smooth potentia...	[math.DS, math.CV]	2020
0806.3026	Distributional Schwarzschild Geometry from non...	In this paper we leave the neighborhood of t...	[physics.gen-ph]	2018
0807.3139	Weight Reduction for Mod l Bianchi Modular Forms	Let K be an imaginary quadratic field with c...	[math.NT]	2019
0810.5491	Nonequilibrium phase transition in a spreading...	We consider a nonequilibrium process on a ti...	[cond-mat.stat-mech]	2020

Table 1: Full dataset (first 5 rows)

B.2 Expanded Dataset

abstract	category
In this work we consider a question in the c...	math.DG
In this work we consider a question in the c...	math.AP
In this work we consider a question in the c...	math.MG
The Gibbs measure theory for smooth potentia...	math.DS
The Gibbs measure theory for smooth potentia...	math.CV
In this paper we leave the neighborhood of t...	physics.gen-ph
Let K be an imaginary quadratic field with c...	math.NT
We consider a nonequilibrium process on a ti...	cond-mat.stat-mech
In this paper we construct integrable three-...	math-ph
In this paper we construct integrable three-...	math.MP

Table 2: Expanded dataset (first 10 rows)

B.3 Main Category Dataset

abstract	category
In this work we consider a question in the c...	math
The Gibbs measure theory for smooth potentia...	math
In this paper we leave the neighborhood of t...	physics
Let K be an imaginary quadratic field with c...	math
We consider a nonequilibrium process on a ti...	cond-mat
In this paper we construct integrable three-...	math-ph
In this paper we construct integrable three-...	math
In this paper we construct integrable three-...	nlin
In this paper we construct integrable three-...	quant-ph
In this study the numerical performance of t...	physics

Table 3: Main category dataset (first 10 rows)

C Unsupervized Learning Models

C.1 K-Means

C.1.1 Elbow Method for Optimal k

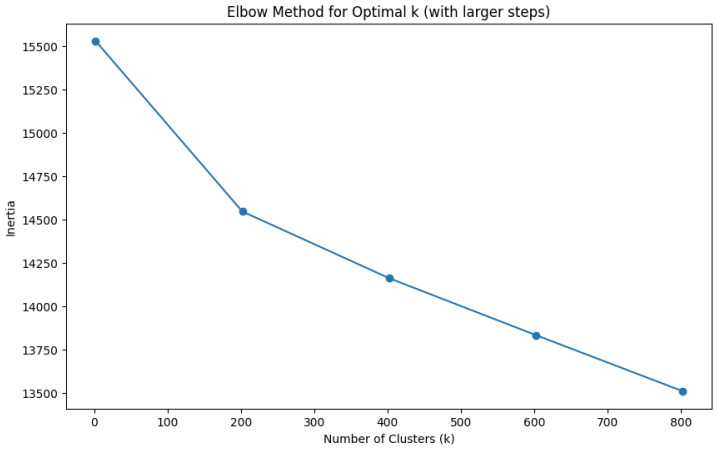

Figure 2: Elbow Method for Optimal k

C.1.2 Clusters visualized - 2D

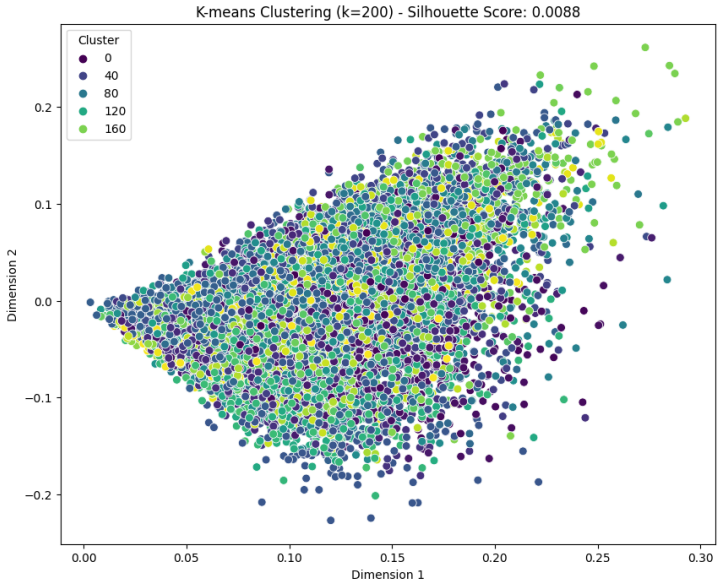

Figure 3: Clusters visualized - 2D

C.1.3 Clusters visualized - 3D

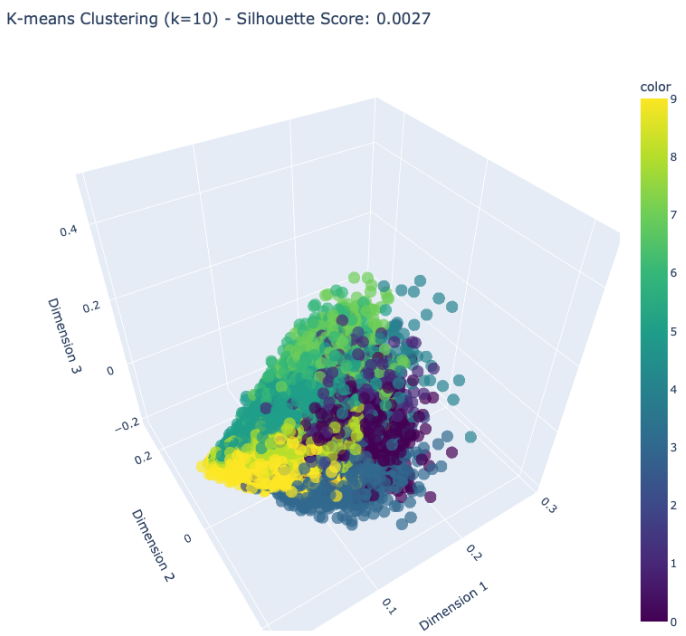

Figure 4: Clusters visualized - 3D

C.1.4 Silhouette Score for Optimal k - Plotted

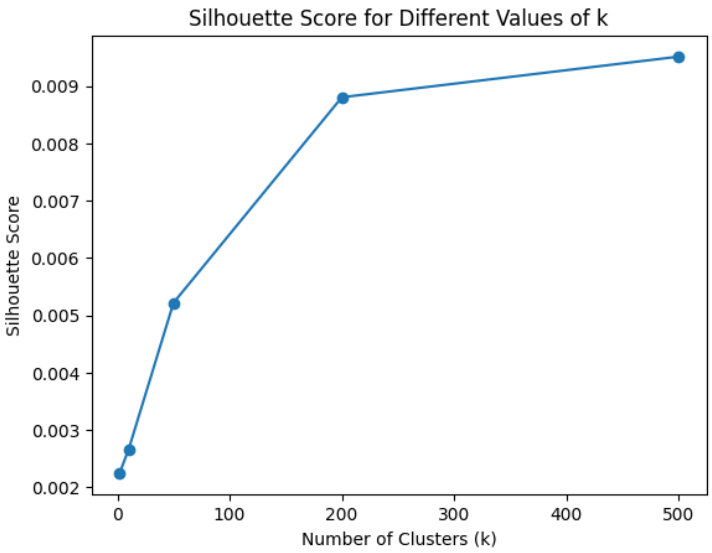

Figure 5: Silhouette Score for Optimal k - Plotted

k	Silhouette Score
2	0.002249724406563622
10	0.002660465057725852
50	0.005218121326725755
200	0.00880186795702813
500	0.009509875425167761

Table 4: Silhouette Score for Optimal k - Table

C.1.5 Silhouette Score for Optimal k - Table

C.2 SVD + DBSCAN

C.3 Explained Variance against Number of Components

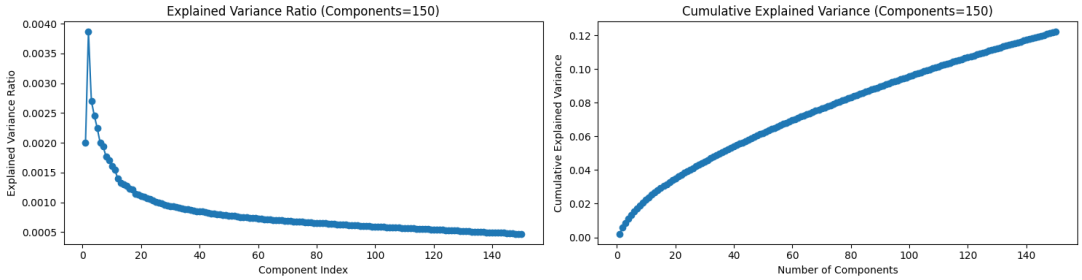

Figure 6: Explained Variance against Number of Components

C.4 DBSCAN Silhouette Scores

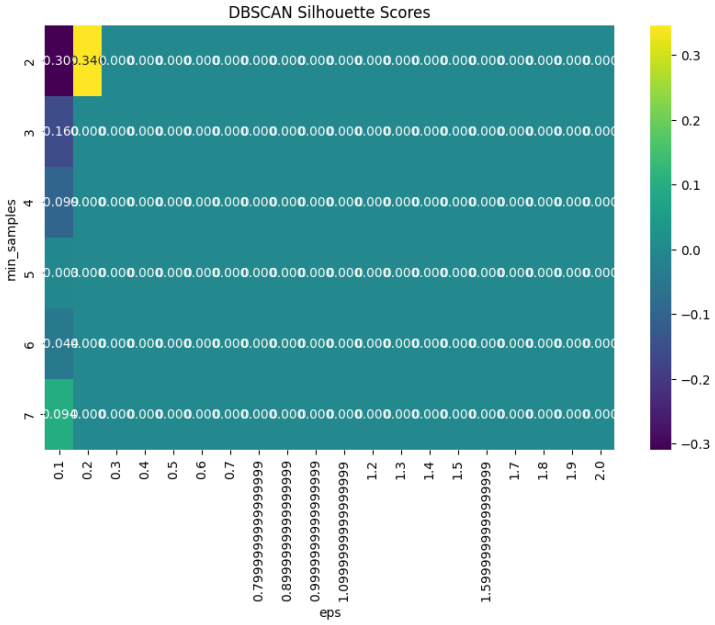

Figure 7: DBSCAN Silhouette Scores - Heatmap

D Supervised Learning Models

D.1 fastText

D.1.1 fastText - Precision against N-grams (Expanded dataset)

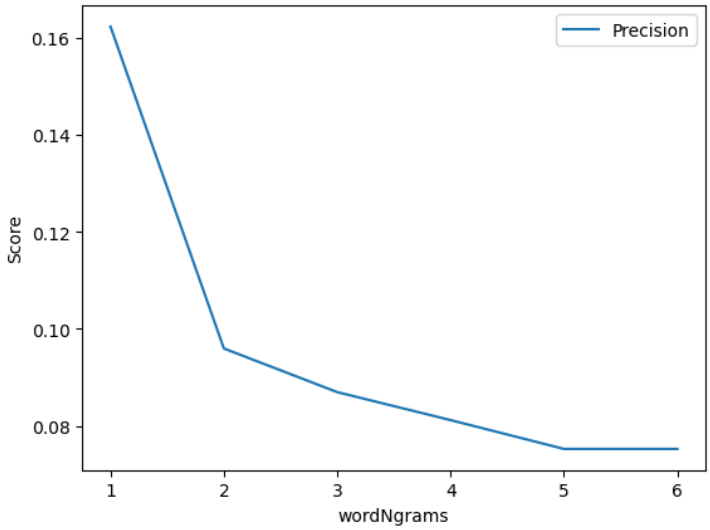

Figure 8: fastText - Precision against N-grams (Expanded dataset)

fastText - Precision against Epochs (Expanded dataset)

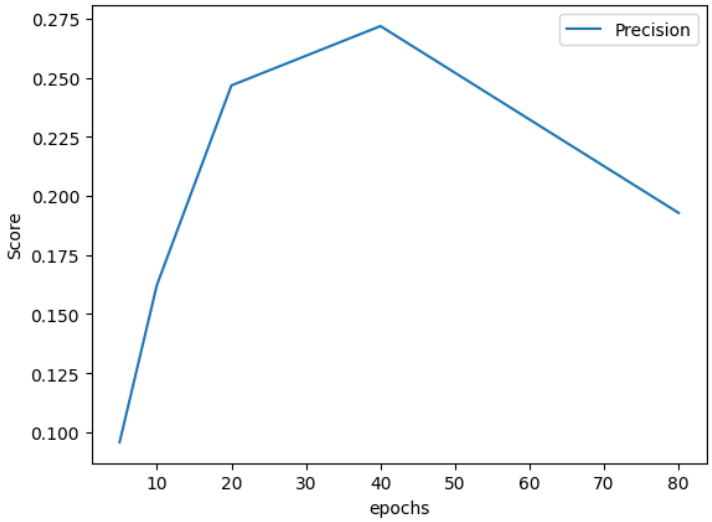

Figure 9: fastText - Precision against Epochs (Expanded dataset)

D.1.2 fastText - Precision against N-grams (Main Category dataset)

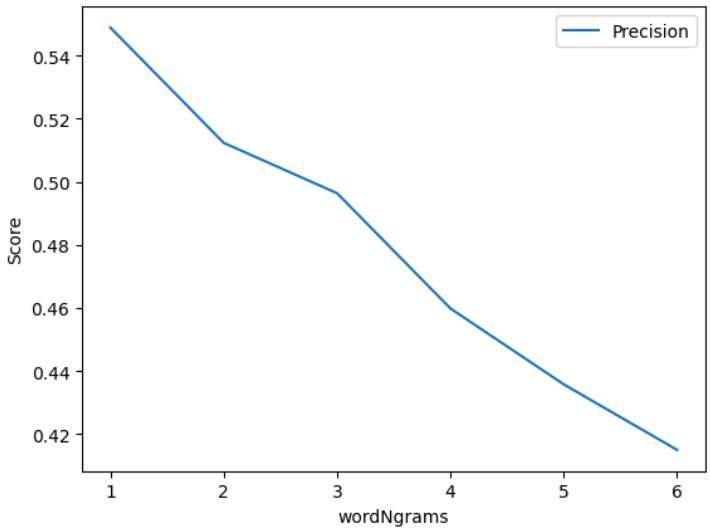

Figure 10: fastText - Precision against N-grams (Main Category dataset)

fastText - Precision against Epochs (Main Category dataset)

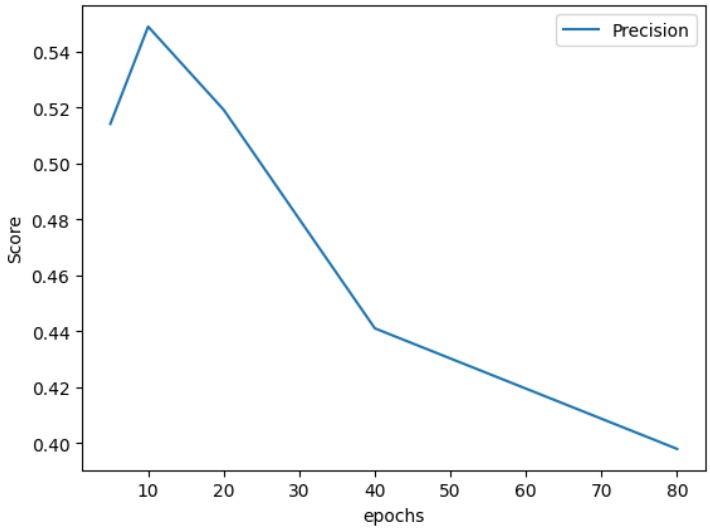

Figure 11: fastText - Precision against Epochs (Main Category dataset)

D.2 ANNOY Model

Time taken to find 10 nearest neighbors for abstract sample 1: 0.8448419570922852

	id	title	abstract	category	year
83	1306.0084	Conditional Expectation of a Markov Kernel Giv...	markov kernels play a decisive role in probab...	[math.ST, stat.TH]	2020
1624	1706.03955	A Note on the Relationship Between Conditional...	two known results on the relationship between...	[math.ST, stat.TH]	2019
588	1606.07282	A review of Gaussian Markov models for conditi...	markov models lie at the interface between st...	[stat.ME, cs.AI, stat.ML]	2020
14433	2010.08864	Markov Neighborhood Regression for High-Dimens...	this paper proposes an innovative method for ...	[stat.ME]	2020
9698	1907.12912	On the estimation of average treatment effects...	we are interested in the estimation of averag...	[stat.ME]	2020
1260	1703.06131	Inference via low-dimensional couplings	we investigate the lowdimensional structure o...	[stat.ME, stat.CO, stat.ML]	2018
8932	1906.00198	nprobust: Nonparametric Kernel-Based Estimatio...	nonparametric kernel density and local polyno...	[stat.CO, econ.EM, stat.ME]	2019
13322	2006.06560	Generalization error in high-dimensional perce...	we consider a commonly studied supervised cla...	[stat.ML, cond-mat.dis-nn, cs.LG, math.ST, sta...	2020
1733	1707.02331	Shrinkage Estimation Strategies in Generalized...	in this study we propose shrinkage methods ba...	[math.ST, stat.TH]	2020
1008	1612.09413	Permuted and Augmented Stick-Breaking Bayesian...	to model categorical response variables given...	[stat.ME, stat.ML]	2018

Figure 12: Sample 1 - ANNOY Model