Prediction of Research Impact : A Case Study for Nanotechnology

Raisul Islam (raisul@stanford.edu)
Patrick Tae (ptae@stanford.edu)
Electrical Engineering, Stanford University

***Abstract*—Classifying research and predicting citation counts is an interesting issue of relevance due to the rapidly growing volume of scientific literature. We attempt to predict research paper citation counts based on simple text features in the title and abstract, using a dataset of papers in the area of nanotechnology. The results indicate that limited success is possible using this approach but ideally more sophisticated features must be processed out of the text to generate more effective predictions. We also successfully identify major categories of research by implementing k-means clustering of the papers.**

I. Introduction

As we live in an era of exponential growth in technology, research areas in the technological sciences are also becoming increasingly diversified. Every day, an overwhelming number of new papers, ideas, or applications are released. As a result, investigating research areas of interest and understanding trends in certain fields has become a challenging task.

The goal of this work is to predict the impact of a research project. We propose to make this prediction using the keywords related the title of the research papers or project, along with the abstract. An effective measure of research impact is the number of citations that a research paper gets over time. So we used two machine learning algorithms, Gaussian discriminant analysis (GDA) and softmax regression, to predict the number of citations a research paper will get given the keywords associated with the title along with its abstract. This could give a prospective researcher some sense of the popularity of a topic, which will allow for informed decisions about research topics to investigate. Alternatively, this tool could help big R&D groups that wants to allocate limited resources more efficiently.

II. Previous Work

There have been various previous works exploring the relationship between papers and citations. Some provide systems for citation recommendations, intended for use by researchers looking for relevant papers to cite in their own work[1-3]. Others attempt to actually predict a paper's total citation count[4] or change in citations over time[5], which are of more direct relevance to this project.

These previous attempts at citation prediction have used various models such as linear regression, k-nearest neighbors, support vector regression, and classification and regression trees[4]. They have also used a variety of features such as the paper's focus content, breadth, publication date, authors, and others. These methods have been reasonably effective at predicting paper citation counts over time. For this work, due to the limitations of our dataset and for simplicity, we have chosen to use papers' title and abstract text as features. We explore a discriminative algorithm, softmax regression, and a generative model, GDA, to attempt classification of a paper's future citations, along with k-means clustering to more broadly be able to categorize the papers.

III. Dataset

We picked the top three journals in the area of nanotechnology on Google Scholar: Nano Letters, ACS Nano, and Nature Nanotechnology[6]. These were ranked based on h-index, which is defined as the number of papers *n* from a journal that have at least *n* citations. The papers contributing to the h-index for each of these journals make up our dataset, which is 523 papers in total. From this list of papers, we extract the words in the title, number of citations, and year of publication, along with the abstracts (for the Nano Letters papers). The text from the titles and abstracts make up the features, along with the publication year. The year and number of citations give us the metric for popularity, citations per year, which we are trying to predict. The input text must be processed to be more useful and usable, as shown in figure 1. The data is represented with a vocabulary of unique root words that appear in the list of titles/abstracts. The list of titles/abstracts is then represented as a list of pointers to the vocabulary, and there is an associated list of corresponding citations and publication years.

Input Raw Data:

1.)	Title: {**Fracture of Silicon Nanoparticles During Lithiation**} -> x Citations: **# of citations / Age in years** -> y
2.)	-Remove non-alphanumeric characters, replace with spaces. -Make all characters lowercase, split on whitespaces to form a vector of keywords. **'fracture' 'of' 'silicon' 'nanoparticles' 'during' 'lithiation'**
3.)	-Remove prepositions/small unimportant words **'fracture' 'silicon' 'nanoparticles' 'during' 'lithiation'**
4.)	-Find root substring of each word by comparing to the vocabulary. If it is a new word, it is added to the vocabulary. **'fracture' 'silicon' 'particle' 'during' 'lithiation'**

Figure 1:
Steps for processing the raw text data from the paper titles and abstracts. The list of words removed in part 3 was obtained by inspection, and includes: {'from', 'for', 'and', 'a', 'as', 'with', 'in', 'at', 'of', 'on', 'the', 'can', 'be', 'by', 'how', 'by'}

IV. Methods and Algorithms

A. GDA Theory

Training:

The input vector is represented as

$$x^{(i)} = \left\{\eta_1^{(i)}, \eta_2^{(i)}, \dots\dots, \eta_{n^{(i)}}^{(i)}\right\}^T \tag{1}$$

where $\eta_j^{(i)}$ is the position in the vocabulary of the j^{th} word in the i^{th} title. The probability that a given word represented by the position η in the vocabulary appears in the input vector, given that the title is assigned to the bin b, is given by

$$p(x|y=b) = \Phi_{\eta|y^{(i)}=b} = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n_i} 1\left\{x_j^{(i)} = \eta \wedge y^{(i)} = b\right\} + 1}{\sum_{i=1}^{m} 1\{y^{(i)} = b\} n_i + |V|} \tag{2}$$

Also, the probability that y is in bin b is given by

$$p(y=b) = \Phi_{y=b} = \frac{\sum_{i=1}^{m} 1\left\{y^{(i)} = b\right\}}{m} \tag{3}$$

These two probability distributions constitute the parameters for classifying a new test sample.

Testing:

For each test vector we calculate the posterior probability, giving us a probability distribution for all the bins. For any bin b,

$$p(y=b|x) = \frac{p(x|y=b)p(y=b)}{\sum_{j=1}^{k} p(x|y=j)p(y=j)} \tag{4}$$

Finally we classify according to the following equation:

$$y_{test}^{(i)} = \begin{cases} \underset{y\in\{1,2,3,\dots,k\}}{\text{argmax}}\ p(y|x), if\ max(p(y|x)) > 0.5 \\ round\left(\sum_{j=1}^{k} jp(y=j|x)\right), else \end{cases} \tag{5}$$

B. Softmax Regression Theory

Training:

The input vector corresponding to the i^{th} title is $x^{(i)} = \{x_j^{(i)}\}$, where $x_j^{(i)}$ is the j^{th} component of the title vector i, i.e. $x_j^{(i)} = 1\{j^{th}$ word in the vocabulary appears anywhere at least once in the title vector $i\}$. The conditional probability of y being in a certain bin given a specific $x^{(i)}$ is given by

$$p\left(y=l\middle|x^{(i)};\theta\right) = \frac{e^{\theta_l^T x^{(i)}}}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}} = \phi_l \tag{4}$$

Here, $y \in \{1,2,3,\dots,k\}$ and k is the number of bins. We consider the parameter θ such that for each possible value of y, there is a corresponding theta vector, defined as

$$\theta = \begin{bmatrix} | & | & \dots & | \\ \theta_1 & \theta_2 & \dots & \theta_k \\ | & | & \dots & | \end{bmatrix} \tag{5}$$

Here, θ_l = column vector corresponding to $y = l$.

Finally the update equation for θ_l is given by

$$\theta_l := \theta_l + \alpha y^{(i)}\left(1 - h_{\theta_l}\left(x^{(i)}\right)\right)x^{(i)} \tag{6}$$

with $h_\theta\left(x^{(i)}\right) = \begin{bmatrix}\phi_1 \\ \phi_2 \\ \vdots \\ \phi_k\end{bmatrix}$. Therefore, $h_{\theta_l}\left(x^{(i)}\right) = \phi_l$ given in equation (4). After the training we obtain the theta vector, which is used for testing.

Testing:

For each test vector $x_{test}^{(i)}$, we calculate the corresponding probability distribution $h_\theta\left(x_{test}^{(i)}\right)$. Then, we look for the highest entry in this vector. If this entry is higher than 0.5 we assign the test vector the label corresponding to the index of the highest entry i.e. the bin number from {1,2,3,…,k}. However, we later observed that for some test samples, no one entry is above 0.5. In that case in order to minimize the mean error, we assign the output to be the expectation value of the indexes rounded to the nearest integer. This is formulated below:

$$y_{test}^{(i)} = \begin{cases} \underset{index\ of\ h_\theta}{\text{argmax}}\left(h_\theta\left(x_{test}^{(i)}\right)\right), if\ max\left(\left(h_\theta\left(x_{test}^{(i)}\right)\right)\right) > 0.5 \\ round\left(\sum_{j=1}^{k} j\phi_j\right), else \end{cases} \tag{7}$$

Performance Metric:

The raw performance of the classification is given by accuracy, defined as

$$\%Accuracy = 100 \times \frac{\sum_{i=1}^{m} 1\left\{y_{true}^{(i)} = y_{test}^{(i)}\right\}}{m} \tag{8}$$

Here, $y_{true}^{(i)}$ is the true class of $x^{(i)}$ obtained after binning the scaled citation (citation per year) data and $y_{test}^{(i)}$ is the predicted class. However, the true prediction is given by the predicted scaled citation, which we assign to be the center of the bin corresponding to the predicted class. So the mean error is given by

$$E = \frac{\left\| C_{true} = c_{y=b} \right\|_2}{m} \quad (9)$$

Here, C_{true} is the true citations per year for the test set and $c_{y=b}$ is the predicted citations per year for the same set, which is essentially the center of the bin *b*.

C. K-means Clustering Theory

For k-means clustering, we seek to categorize the papers into *K* clusters, each defined by *W* words, with word *w* in cluster *k* denoted C_{kw}. We define a metric of similarity, S_{ki} between a cluster k and input $x^{(i)}$ with *J* unique words:

$$S_{ki} = \sum_{w=1}^{W} \sum_{j=1}^{J} 1\{x_j{}^{(i)} = C_{kw}\} \quad (10)$$

So the similarity metric is simply the number of shared words between an input and the cluster's defining words.

After randomly initializing the clusters, the algorithm itself proceeds as follows: Put each input into the cluster with the highest similarity to it. Then redefine the words in the cluster to be the *W* most common words appearing in the inputs in that cluster. Continue iterating until the *W* words for each cluster no longer change between iterations.

We also define another metric, E_k, which measures the average similarity between a cluster and its constituent words:

$$E_k = \frac{1}{n_k} \sum_{i_k=1}^{n_k} S_{ki_k} \quad (11)$$

Here, i_k indicates that the sample is in cluster *k*, and there are a total of n_k such samples. This metric will be useful for selecting the number of clusters, discussed in more detail below.

V. Results

A. Citation Prediction

We ran both GDA and softmax regression on our set of titles, with 70% of the samples being randomly put into a training set and the remaining 30% being used for testing. The α parameter for softmax was fixed at 1e-3, as this was found to give the best results in general. The size of the bins that the algorithms classify into were also swept as a parameter to see how that affects classification accuracy and mean error. We also used a simple random guess algorithm to establish a performance baseline to judge the other algorithms. The random algorithm chose a bin by sampling from the distribution of citation counts per year observed in the data, which is shown in figure 2.

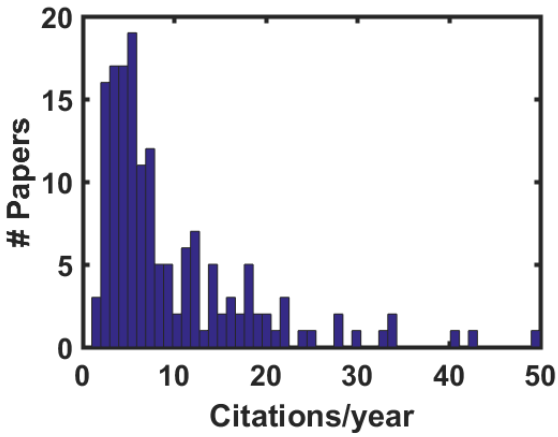

Figure 2:
Distribution of paper citation counts per year for our dataset.

Classification accuracy and mean error, defined in equations 8 and 9, respectively, are our performance metrics of interest. Figure 3 shows these metrics for the three algorithms plotted against the number of classification bins. The classification accuracy decreases with the number of bins, which is to be expected because having more bins makes it more difficult to pick the correct one. Of course, having a higher accuracy with fewer bins doesn't necessarily mean that fewer bins is better from a predictive standpoint because fewer, bigger bins imply a larger possible range of values within a bin, meaning that correct predictions are less useful.

For this reason, the mean error is a more useful metric for judging performance. As seen in figure 3, it is large when there are very few bins but quickly drops to a relatively constant value, which may then increase slightly with the number of bins. The large error for few bins is likely due to our linear binning strategy combined with the skewed distribution of citations. Since the mean error is calculated as the difference from the center of the bin to the true value, if there are no bins with centers near the peak of the probability distribution (as is the case with very few linearly spaced bins over a skewed data distribution), the error will be large.

In terms of algorithm performance, it is clear that GDA performs the best, with a definite improvement over random guessing. Softmax does somewhat worse, with performance similar to or slightly better than random guessing. Table 1 gives the training and testing errors for each of the algorithms, for 20 bins. The small training error and large test error of softmax suggests that overfitting is the reason for its inferior performance. Thus, a logical step for improving softmax for this application in the future would be to add some kind of regularization term.

While the algorithms have performed better than baseline random guessing for classifying titles, the overall performance still leaves much to be desired. We attempted to use abstract text as an additional feature to improve the prediction, since an abstract contains far more information than a title. We ran the same algorithms on the set of abstracts from one journal (Nano Letters), treating the abstract text in the exact same as

the title text in terms of data processing and algorithm execution. Unfortunately, we found that the performance for classifying these abstracts was substantially worse compared to the titles alone, and was in general similar to or worse than random guessing. A possible explanation for this is that the number of words in an abstract is much larger, requiring far more training samples to generate reasonable probability distributions for them, and our dataset (consisting of less than 200 abstracts) may be far too small. It is also possible that the statistics from parsing words in this way provide very little predictive power, so more advanced techniques, from natural language processing for instance, may be needed to get actual ideas and themes from the abstracts to be use as features.

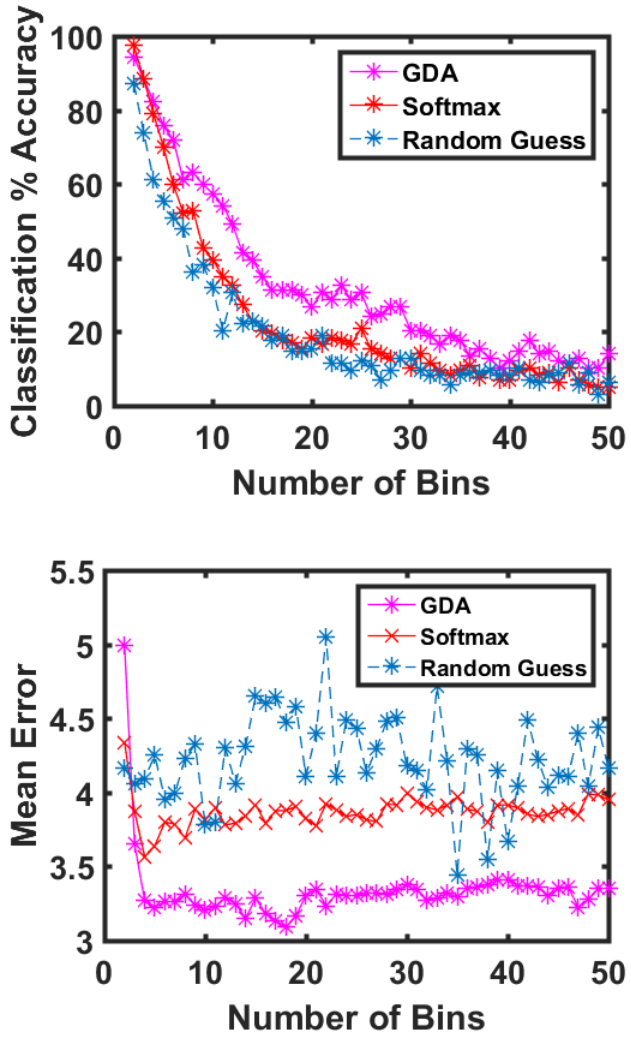

Figure 3:
Classification accuracy (top) and mean error (bottom) for GDA, softmax, and random guessing vs. number of bins.

Algorithm	Train Error	Test Error
GDA	1.81	3.31
Softmax	0.996	3.83
Random Guess	4.28	4.28

Table 1:
Train and test errors for each of the three algorithms, using titles as features with 20 bins.

B. *K-means Clustering*

We decided that the number of words used to represent a cluster should be between 5 and 10, since this corresponds to the typical length of titles in our dataset after processing. We ultimately settled on 8, although any number in this range should work reasonably well. For selecting the number of clusters, we swept the cluster number from 2 to 10 and looked at the average E_k of the clusters under each condition to see how well the cluster titles represent their cluster in general. The results of this are shown in table 2. It is clear that having more clusters results in higher average similarity between the cluster title and its constituent titles, because each cluster has fewer samples and can match its words more closely with those samples. However, it is also clear that there are diminishing returns in this regard as the number of clusters continues to increase. So we chose to use 6 clusters, as this appears to be the point where adding additional clusters has a minimal impact on E_k.

After running k-means clustering on the titles with 6 clusters, we obtain clusters similar to the ones shown in table 3. We can see that the words defining each cluster are clearly grouped according to more specific research areas, and using a little knowledge of the field we can assign labels to the clusters, identifying each of these areas. So it appears that k-means can effectively group paper titles into their general broad categories, provided that the number of clusters and words are chosen appropriately. It should be mentioned that repeatedly running this algorithm will yield slightly different results each time, and what is seen in table 3 are the results for a typical representative run. Do to the large number of words in the vocabulary, the dimensionality of the space if very large and also discretized, which leads to many local minima that the clustering algorithm can fall into. Thus, it is important to run the algorithm multiple times to ensure that the results are consistent, or that the final similarity in the clusters satisfies some threshold.

Number of clusters	Average E_k
2	16.6
3	20.1
4	22.2
5	22.6
6	24.7
7	24.7
8	25.5
9	26.5
10	26.2

Table 2:
Average E_k for the clusters obtained after k-means clustering using different numbers of clusters, with 8 words identifying each cluster.

1: High Performance	**2: Solar Cells**	**3: Transistor Technology**
graphene	solar	layer
high	cells	mos2
carbon	graphene	transistor
performance	layer	effect
super	oxide	particle
nanotube	electron	silicon
structure	sensitized	field
hybrid	perovskite	single
4: Battery Technology	**5: Drugs / Medical**	**6: General Materials**
graphene	plasmon	electrode
particle	surface	ultra
cation	sensor	mos2
dimensional	particle	structure
based	nanorod	material
material	therapy	crystal
micro	inter	super
cells	cancer	sheet

Table 3:
List of clusters and associated words resulting from running k-means clustering with 6 clusters and 8 words per cluster. The bold label at the top was manually assigned to describe the type of research represented by the cluster.

VI. Conclusion and Future Work

We have attempted to predict the number of citations that a research paper will receive in the future by looking at the title keywords as well as the abstract. We used Gaussian discriminant analysis and softmax regression to generate predictions of citations per year for a set of papers in the general area of nanotechnology, and focused on the metrics of classification accuracy and mean error. When using only titles as features to classify, we found that GDA produces the best predictions across all classification bin sizes, while softmax is only slightly better than random guessing over the data distribution. When trying to make predictions using abstracts, we found that the algorithms actually do worse and don't provide a meaningful improvement over random guessing. This indicates that abstracts needed a separate form of data representation and more features like interdependence of the keywords need to be incorporated in order to predict accurately.

We also implemented k-means clustering to attempt categorizing the papers based on their titles. The algorithm was developed to maximize the similarity between the words of titles in the same cluster. The number of clusters was chosen by sweeping this parameter and considering the resulting average similarity between titles and clusters. Using 6 clusters with 8 words per cluster we were able to identify reasonable clusters of titles corresponding to relevant subcategories of research involving nanotechnology.

Moving forward, it is clear that doing citation prediction using only title text is difficult do to the very limited amount of information in a title. However, expanding the text features to include the abstract is not likely to help much because counting words alone doesn't offer enough predictive power. More sophisticated features need to be extracted from the text, such as major ideas, topics, breadth of work, relevant results, and conclusions. These features should possible to extract, at least to some extent, with the help of natural language processing techniques. Other features to consider would be author identities along with their prominence and network connections within the field. Considering these more sophisticated features could also help improve the performance of the clustering algorithm by creating a more accurate/meaningful similarity metric to be used for constructing the clusters. Another possibility for future work is to use to clusters obtained from k-means and run classification algorithms such as GDA or softmax on the individual clusters. Training on individual clusters of highly correlated papers may lead to better overall performance. This would certainly require a larger dataset as well because our current dataset is not really large enough and this issue would be exacerbated by dividing the data further into clusters. Finally, in this work we have found a relatively weak classifier for citation prediction, which does slightly better than random guessing. Boosting can take a weak classifier and turn it into a strong classifier by changing weights, so this would be another interesting option for improving on the existing work.

VII. References

[1] He, Qi, et al. “Context-aware citation recommendation.” *Proceedings of the 19th international conference on World wide web*. ACM, 2010.

[2] Ren, Xiang, et al. “Cluscite: Effective citation recommendation by information network-based clustering.” *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014.

[3] Yu, Xiao, et al. “Citation Prediction in Heterogeneous Bibliographic Networks.” *SDM*. Vol. 12. 2012.

[4] Yan, Rui, et al. "Citation count prediction: learning to estimate future citations for literature." *Proceedings of the 20th ACM international conference on Information and knowledge management*. ACM, 2011.

[5] Manjunatha, J. N., et al. "Citation prediction using time series approach kdd cup 2003 (task 1)." *ACM SIGKDD Explorations Newsletter* 5.2 (2003): 152-153.

[6] Google Scholar Metrics, Top Publications – Nanotechnology, URL: https://scholar.google.com/citations?view_op=top_venues&hl=en&vq=eng_nanotechnology