Repeatability and Self-Tuning in Sentiment Analysis of Scientific Citation

Huy Tu
North Carolina State University
Raleigh, North Carolina
hqtu@ncsu.edu

ABSTRACT

Sentiment analysis of citations within scientific papers have been studied to more appropriately indicating the quality of published papers in comparison to the outdated quantitative evaluation approach basing on the citations' frequency. Approximating the popularity, the context, and the impact of the published research are commonly referred to as bibliometric measurements. In this study, we shall review state of the art sentiment analysis algorithms that helps incorporated that qualitative aspect to the bibliometrics. Moreover, we will focus on the improvement of result analysis of those state of the art algorithms while making it more repeatable by proposing empirical software science approaches such as dimensionality reduction, synthetic minority oversampling, and parameter tuning.

***Keywords* -** Citation Sentiment Analysis, Classification; Differential Evolution, Parameters Tuning; Dimensionality Reduction, Repeatability.

1 INTRODUCTION

Bibliometrics are measures of popularity and the impact of the published research which are important and necessary to keep the research community alive. The old-fashioned quantitative citation bibliometrics include Hirsh-index, the g-index [8], and PageRank [16]. The moden qualitative NLP-based bibliometrics include supervised learning of polarity and purpose of citation classification. Citation polarity classification through sentiment analysis aims to determine opinions, emotions, and attitudes of the specific granularity of the text region. It can be conducted at three levels of document-level, sentence-level, and context-level.

Text analysis, specifically sentiment analysis dataset tend to be sparse or multi-dimensional data (words vector within a document with relation to documents across the corpus or multiple corpuses). Specifically, the specific study of sentiment analysis in scientific citations can lead to availability of large size dataset with numerous dimensions in which the measurements that best reflect the dynamics of our system is unknown. Sometimes dimensions are recorded and created more than we actually need, not mentioning the efforts being put into the work to generate the dimensions. Finally, in order to solve the perspective problem, high praised off-the-shelf state-of-the-art models such as Neural Networks, Support Vector Machine, and k-Nearest Neighbors tend to be considered even when they are naturally complex, thorough, and slow.

In the real world, the produced model has to be rerun multiple occasions in order to satisfy the additional data or data's dimensions/features, updates on implementation architecture, or consistent performance and results. Moreover, the dataset's size and complexity would be significantly larger which leads to repeatability become a challenge in the field, especially with the complex, thorough, and slow state-of-the-art models such as NN, SVM, and k-NN.

There were many efforts in improving data's features quality such as accurately defining citation contexts and pre-processing such as typed dependencies generation, part-of-speech tagging, type of references identification, or number of grams (Garzone, 1997; Nanba et al., 2000; Pham et al., 2003; Teufel etal., 2006; Angrosh et al., 2010; Athar, 2011 2012) [7] . However, from the early analysis, the dataset is highly skewed or imbalance which leads to little knowledge for the model to perform well across all classes prediction. Moreover, there were little to no efforts in optimizing model's parameters but simply applying the off-the-shelf state-of-the-art models to solve the problem. Yet, fine tuning of parameters have been studied to improve the result analysis by optimizing the textual data and tuning does not need to be costly.

Therefore, this study shall investigate these following research questions:

***RQ 1:* What approach to take for dealing with growing dimensions of the dataset? By doing so, how does it make the model become more repeatable?**
Principal Component Analysis (PCA) to retain the most informative features/dimensions within the dataset while dropping the less informative ones (from 10561 dimensions reducing to 1000). Both the model and data size are significantly reduced with greatly faster CPU time.

***RQ 2:* Is the results from tuning the baseline algorithms with parameters tuning would outperform the results achieved by the standalone baseline algorithms from previous work?**
The results of SVM's performance on dataset after PCA according to F1-Micro and F1-Macro higher than the previous work's report on the same metrics. The difference are also statistically significant. However, the result does not stay consistent with all other learners' results.

***RQ 3:* What are the trade-off for tuning and are they acceptable?**
By looking at other criteria such as time and storage, the decision of how worthwhile it is to apply self-tuning with PCA preprocessing on the model for this study instead of off-the-shelf model is uncertain for future exploration.

The rest of the paper is organized as follow: [2] discusses the related work in sentiment analysis of scientific citations and the evaluation criteria motivation to incorporate repeatability and self-tuning into state-of-the-art models, [3] expands in details on the dataset overview and the experimental design of the project, [4] reviews the results according to the criteria and research questions,

[5] analyzes potential threats to validity of this work, and [6] concludes the work and the lessons while mapping out the plan for future work.

2 BACKGROUND AND MOTIVATION

2.1 Polarity Analysis of Scientific Citation

Bibliographic citations have kept the research community alive. Iorio et al [12] have expressed the importance of bibliographic citations as they are tools for:

- *disseminating research* - with references as directional edge to point to motivational background, related work, data sources, and methods coming from various publishing platforms.
- *exploring research* - the network of citations (papers or authors or journals are presented by the nodes and the edges representing the referencing act) can be utilized as readable data for searching, visualizing, filtering, and aggregating for interesting problems such as knowledge transfers across domains, scientific collaboration, and the scholarly disciplines map.
- *evaluating research* - characterizing and quantifying the importance and impact of the work/research through the nature of the purposes those citations and not only the mere existence of those papers.

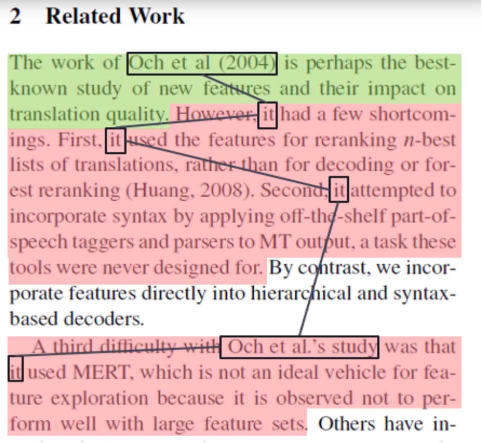
In this paper, we will focus only at the third important aspect of bibliographic citations, evaluating research. Evaluating research's impact should be optimized for fairness and objectiveness through both quantitative and qualitative evaluations. Only considering one without the other aspect of the evaluation would be misinformed, narrow-minded, and problematic.

- For quantitative aspect without qualitative aspect, the nature of citation itself would not be considered and all the citations would be taken in as they carry equivalent weights when they should not. For example, the referencing act of crediting should be counted differently for the referencing act of criticizing.
- For qualitative aspect without quantitative aspect, the interest of other researcher in it and the popularity of the work [1] would be missing. Moreover, there would be no statistics for significance and effect insights from the qualitative evaluation.

The old-fashioned quantitative citation bibliometrics include Hirsh-index, the g-index [8], and PageRank [16]. The moden qualitative NLP-based bibliometrics include supervised learning of polarity and purpose of citation classification.

Citation purpose classification is the studying of the motive behind citing that work or research. In 1977, Spiefel-Rosing et al. [19] formerly suggested 13 categories for citation purpose. In 2006, Teufel et al. [20] adopted 12 categories from Spiegel-Rosing's work that can be appropriately grouped to four types: weakness, contrast (4 categories), positive (6 categories), and neutral. In 2013, from those previous work, Abu-Jbara et al. [1] condensed the previous's quantity of categories down to only six categories of: criticizing, comparison, use, substantiating, basis, and neutral(other).

Citation polarity classification through sentiment analysis aims to determine opinions, emotions, and attitudes of the specific granularity of the text region. It can be conducted at three levels of document-level, sentence-level, and context-level. We will focus more on citation polarity classification on context level for this work. The document-level and sentence-level of sentiment analysis have been incomplete and defective due to how citation region can be varied, how multiple citations can be included in one sentence while having different sentiment tags, and how the sentiment can be hidden in the context level. For example:



2 Related Work

The work of Och et al (2004) is perhaps the best-known study of new features and their impact on translation quality. However, it had a few shortcomings. First, it used the features for reranking *n*-best lists of translations, rather than for decoding or forest reranking (Huang, 2008). Second, it attempted to incorporate syntax by applying off-the-shelf part-of-speech taggers and parsers to MT output, a task these tools were never designed for. By contrast, we incorporate features directly into hierarchical and syntax-based decoders.

A third difficulty with Och et al.'s study was that it used MERT, which is not an ideal vehicle for feature exploration because it is observed not to perform well with large feature sets. Others have in-

Figure 1: Difficulties for Sentiment Analysis for Contextual Citations [4]

In the figure 1 above, the citation of Och et al from "Smorgasbord of Features for Statistical Machine Translation" paper firstly included 'best known study' [11], making this citation positive. However, if we read the following sentences after this citation, we find that this citation is being referred in the next 5 sentences anaphorically like the word 'it' and other such phrases which criticizing the original referenced work negatively which makes this citation sentiment difficult to judge.

Specifically, there are still many difficulties and open research opportunities within the existing models for citation sentiment analysis [3]:

- Sentiment is often subtle or even hidden in citation within research community. Negative polarity is often embedded contrastively.
- Sentiment of the citation within sentences are often neutral.
- Diverse variation of sentiment lexicon or sentiment carrying scientific and/or technical terms/phrases (and specifically from the author's research area). Some are longer in length (such as âĂIJstate of the artâĂİ) which suggests that considering higher n-grams would be useful.
- From a single clause level to multiple paragraphs level, the region of influence of citations differ.

Sentiment polarity varied within context which inherits the cognitive and social values from the research community's culture. It is definitely an interesting problem that has been studied as a subject of scholarly analysis in the research community. From the previous work, they can be grouped into two main type of sentiment analysis research. The old-fashioned one focused on rule-based schema based on a reconstructed decision tree classification having pre-defined cue words and phrases set to classify extracted citation scope (Garzone, 1997; Nanba et al., 2000, Pham et al., 2003) [7]. The more advanced one incorporated state of the art machine learning models such expert knowledge of lexicon (scientific and technical terms) or phrases (cues) (Angrosh et al., 2010; Teufel et al., 2006; Athar, 2011 & 2012) [7]. Those machine learning based classifiers included IBk k-NN (k- Nearest Neighbors), support vector machine (SVM), and CRF. The summary of the previous efforts is recorded in the Table 1 based on their features and classifier.

Table 1: Comparison of Citation Purpose & Polarity Analysis Schemas [7]

Work	Features	Classifier
Teufel et al. 2006	Cue phrases Verb tense/voice Modality Location (paper/paragraph)	IBk (k-NN)
Angrosh et al. 2010	Generalization terms (Lexicon) (Prev.) Sentence has citations	CRF
Dong&Schafer 2012	Cue words Boolean and weight POS-tag	SMO BayesNet NaiveBayes
Athar 2012	four-class anotation 1-3 grams Scientific lexicon POS-tag Contextual Polarity Dependency Structure Sentence Splitting (removing) Negation	SVM
Aju-Jbara et al. 2013	Reference Count Reference Separation Cue words Self-Citation Contains 1st/3rd Person Pronouns Contrary Expression Dependency Relations Negation	SVM

2.2 Evaluation Criteria

Evaluating the performance should not be based solely on ground truth based metrics (accuracy, area-under-curve, etc), but also important criteria such as context-aware, self-tuning, anomaly detection, incremental learning, self-tuning etc. For this work, repeatability and self-tuning will be focused on. In order to support those important criteria, it is possible that some compromises have to be made.

Learnability and Repeatability of the Results: The availability and weight of data have grown exponentially over time in this 21st century which is difficult when you need to develop a model that can be applied to most data appropriately. Therefore, the learnability and repeatability of the results is important to improve and reproduce results to confirm performance and adjusting the developed model is an continuous act. The small memory footprint (RAM, disk, CPU, and GPU) models/algorithms are appealing such as Mini Batch K-Means and Naive Bayes.

Self-Tuning: Self-tuning or auto-tuning refers to the aspect of able to optimize itself (by adjusting it's own parameters) which is whatever models and software we are using. Self-tuning aspect helps to satisfy the objective function by maximizing or minimizing the appropriate requirements. Examples of self-tuning include increase of analysis results, maximization of efficiency, or error minimization. The self-tuning method is based on finding the optimal set of gains from a pre-generated training set for a further selection of the best seeds through a membership function.

2.3 Critique

Most state of the art machine learning algorithms for sentiment analysis are stand-alone and stand-alone tools in scientific citation analysis. There have been a lot of efforts putting into detecting citation region and defining/analyzing useful structural features (in word-level, sentence-level, and context-level). The author(s) prepared and cleaned the data as the citation text for tagging, parsing, and transforming before applying the classifier. Those machine learning models include IBk k-NN, SVM. Those work are valuable as discussed above, yet there are several concerns with that approach:

- Abundant and unnecessary dimensions are recorded leading to noises and variance, affecting the model's performance.
- The imbalance class nature of the research community where majority of the citation are objective.
- Off-the-shelf algorithms have parameters that are not tuned to maximize the performance.
- The instability of the prediction method that results in very different outputs with slightly different inputs.

As a result, they potentially miss out the repeatability, reliability, and optimization for the result of the model. Fine tuning of parameters have been studied to improve the result analysis by optimizing the textual data which in this case are citation context using differential evolutionary algorithm. Moreover, Colleta et al. [6] has employ an ensemble method including SVM classifier with cluster ensembles combination which resulted in better classification accuracy. According to Menzies [10], tuning is under-explored for optimization problems.

Consequently, the execution of this essay/plan for the project will strive for employing those ideas into the sentiment analysis of scientific citation problem and combining the dimensionality reduction, minority over-sampling technique, and hyper-parameter tuning method.

3 EXPERIMENTAL DESIGN

The dataset for this study includes manually annotated 7261 citations in the 310 research papers taken from the ACL Anthology from Athar's paper in 2011 [3]. Moreover, there are 209k words/phrases features and 88k typed dependency features in the citation dataset. The overlook of the dataset is shown in the table 2 below. There are more than 295k features along with the imbalance class problem.

Table 2: Overview of the Citation Sentiment Dataset

Sentiment Class	Count	Distribution
Objective ('o')	6276	86.43%
Positive ('p')	742	10.22%
Negative ('n')	243	3.35%

Overall architecture of the model is shown visually in the Figure 2. The dataset and each algorithm based approach can be replaced for reproducibility purpose while maintaining the validity of the study. In summary, the process include:

1. Apply PCA as Dimensionality Reduction to the dataset
2. Split the dataset to train set, tune set, and test set
3. Sampling the train set with modified SMOTE
4. Tuning and Evaluate:
   - Hyper-parameters Tuning the learner (SVM, CART, RF, or k-NN) with Differential Evolution (DE) on train set
   - Cross-validation on tune set
5. Apply tuned model on test set to evaluate

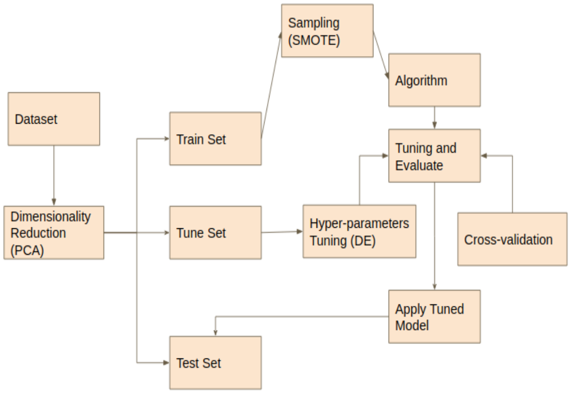


Figure 2: Implementation Architecture

3.1 Dimensionality Reduction

One of the popular approach for dimensionality reduction is PCA. Shlens discussed how Principal Component Analysis provides a roadmap for how to reduce a complex data set to a lower dimension to reveal the sometimes hidden, simplified structures that often underlie it [18]. PCA yields the directions or eigenvectors (principal components) of maximum variance in high-dimensional data and project it onto a smaller dimensional subspace while retaining most of the information (dropping the "less informative" eigenvector-eigenvalue pair). Not only it reduces the search base for solutions but also smoothen out the variances and noises that contaminates the data set only serving to obfuscate the dynamics further.

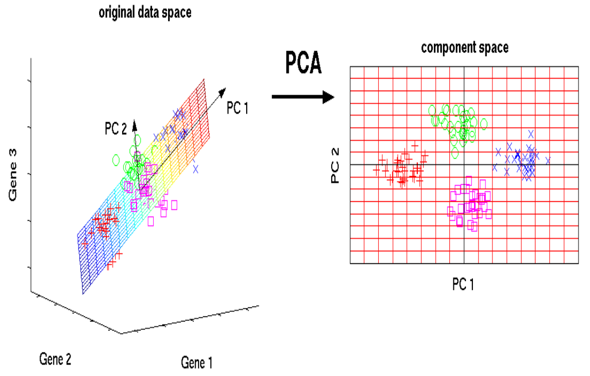


Figure 3: PCA example for three-dimensional gene expression data [17]

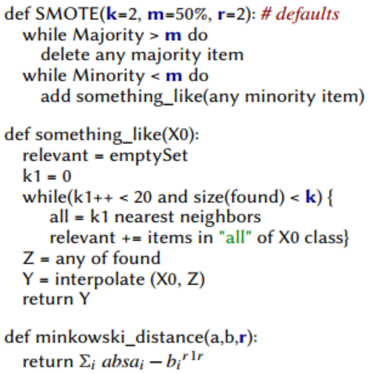
For example, in the Figure 3 above, by applying PCA from the three dimensions on the left, we can identify the two-dimensional plane that optimally describes the highest variance of the data (PC1 and PC2, rotated and presented as a two-dimensional component space on the right) [17]. Athar used 10,561 features in his paper [3] but for this project, the hypothesis is that by applying PCA to reduce the dimension space to 500 and 1000, the storage and time criteria will significantly reduce while achieving reasonably comparable results.

3.2 Sampling

As previously discussed, the current dataset is highly imbalanced the objective class data overwhelmingly dominate the subjective class data (86.43% of the whole dataset). This study plan to handle that by applying modified SMOTE in train data. Modified SMOTE is a combination of over-sampling the minority classes and under-sampling the majority class can achieve better classifier performance (in AUC space) than only under-sampling the majority class or only over-sampling the minority class [2].

For this study, the subjective classes data (positive and negative) will be over-sampled and the objective class will be under-sampled so they three class have approximately similar size (33% for each class data). By doing so, the variance and noise within the majority class data will be reduced while the minority class data distribution will be kept.

The implementation of modified SMOTE can be reviewed in Figure 4 below.



```
def SMOTE(k=2, m=50%, r=2): # defaults
   while Majority > m do
      delete any majority item
   while Minority < m do
      add something_like(any minority item)

def something_like(X0):
   relevant = emptySet
   k1 = 0
   while(k1++ < 20 and size(found) < k) {
        all = k1 nearest neighbors
        relevant += items in "all" of X0 class}
   Z = any of found
   Y = interpolate (X0, Z)
   return Y

def minkowski_distance(a,b,r):
   return $\Sigma_i \; absa_i - b_i{}^{r\,1r}$
```

Figure 4: Modified SMOTE Pseudocode [2]

3.3 Hyperparameters Tuning

No machine learning model work for all dataset and situations, but most machine learning models are predefined with sub-optimal parameters and the built model depends on those parameters and the data to learn. The act of optimization for parameters within a learner can impact the performance in term of the efficiency and result analysis of the learner. Some of the parameters of state of the art machine learning algorithms are below:

- Parameter C to set the amount of regularization of SVM (Support Vector Machine) module in *Scikit-learn*
- Parameter *number-of-trees* to set the quantity of decision trees in a random forest
- Parameter K to set the number of nearest neighbor in kNN (K Nearest Neighbors)

These parameters have similar function within their machine learning models. A small value for parameter C, number of decision trees, and K will generate a simple model with potentially more training errors and a larger value setting for those parameters would result in a more refined model with less training error. However, the improvement of result analysis decreases as the quantitative settings for those parameters increases, i.e. at a certain point the benefit in prediction performance from operations (applying more regularization, learning more trees, and checking more members) will be lower than the cost in computation time for these operations. Moreover, there are abundant considerations or settings for the control parameters (1000+ combinations for several models) that leads to intensive computing power. Efficiency and caution are important aspects in the search for the right settings of parameters to obtain the best performance for the model.

In the field of software engineering, features optimization have not been common till recent [9]. Regarding those previous work on sentiment analysis for scientific citations, with no parameters tuning methods applied, any parameters tuning method would be sufficient to start with. The traditional and popular Grid Search do search for the entire space but they are very slow and may not be the most effective for rerun the algorithm on the dataset repeatedly. Random search algorithms (e.g. differential evolution algorithm) are arguably can outperform Grid search algorithms with efficiency and performance according to Bergstra et al. [5]. Differential Evolution (DE) algorithms are simple to code and have been shown to tune parameters effectively. DE optimizes a problem by maintaining a population of candidate solutions and iteratively creating new candidate solutions by combining existing ones according to a given measure of quality or parameter, and then keeping whichever candidate solution has the best score or fitness on the optimization problem at hand. Moreover, according to Menzies work [10], Grid Search can take to thousands of iterations to converge while the Evolutionary algorithms converge drastically faster in most cases, around 100 iterations.

For this study, the space of parameters combination, the specific parameter and their's values range, are listed below:

Table 3: kNN Tuning Space

Parameter	Tuning Range
n_neighbors	[2, 10]
weights	['uniform', 'distance']

Table 4: SVM Tuning Space

Parameter	Tuning Range
C	[1, 100]
kernel	['linear', 'poly', 'rbf', 'sigmoid']
coef0	[0, 1]

Table 5: Random Forest Tuning Space

Parameter	Tuning Range
max_features	[2, 100]
max_leaf_nodes	[2, 50]
min_samples_split	[2, 20]
min_samples_leaf	[2, 20]
n_estimators	[50, 150]

Table 6: CART Tuning Space

Parameter	Tuning Range
max_features	[2, 100]
max_depth	[2, 50]
min_samples_split	[2, 20]
min_samples_leaf	[2, 20]
n_estimators	[50, 150]

4 RESULTS

***RQ 1:* What approach to take for dealing with growing dimensions of the dataset? By doing so, how does it make the model become more repeatable?**

As discussed above, PCA can reduce the dimension of the dataset by extracting a lower dimensional component space covering the highest variance since the lower variance can be assumed to represent undesired background noise. From our results on reducing to 500 and 1000 dimensions, 1000 dimensions retain more important information while still achieving comparable classifier metrics.

To measure repeatability of the study, we considered three measures 1) Storage space and 2) CPU time. Below we explain each of these measures and their significance:

- Storage space: is the amount of space the data and model take in the memory. Model size is specifically created by each learner after fitting the data occupy the memory which tends to grow depending on the complexity of the model's architecture. Data size tends to be ignored nowadays with accessibility to high power computing tools. Yet, in real world, by doing dimensionality reduction, both data and model size can be significantly decrease. Data and model size are measured in Kilobytes and Megabytes. The storage space is recorded in Table 7 below.

Table 7: Storage

Type	Raw	PCA_1000
Data	154.8 MB	51.8 MB
KNN	900.3 MB	21.5 MB
SVM	173 MB	9 MB
Random Forest	1.1 MB	354.5 KB
CART	78.2 KB	9.2 KB

- CPU time: the amount of time (in seconds) that the learner takes to fit the data and do the scoring on the data. We also recorded the time for tuning to discuss for ***RQ 3*** later. The CPU time is recorded in Table 8 below.

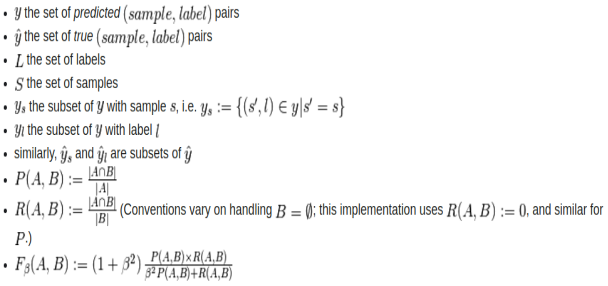
Table 8: CPU Time

Type	Raw	PCA_1000	Tuning
KNN	575.24 s	64.65 s	353.43 s
SVM	332.35 s	36.86 s	246.5 s
Random Forest	7.82 s	4.07 s	225.13 s
CART	72.1 s	0.41 s	353.43 s

From both of the table 7 and 8, the difference is significant after doing PCA to reduce the dimension size to 1000. It reduces 67% of the data size more than 80% for model size (97% for kNN and 94% for SVM). Regarding time, the range is interestingly vast, from twice faster for RF to 175 times faster for CART. Moreover, by using CART model over SVM, we achieved 810 times faster. Even if we decided to tune CART, the time for tuned CART (353.84 seconds) and the time for running SVM on the previous work's dataset (332.35 seconds) are comparable. Yet, are the classifier results also comparable?

***RQ 2:* Is the results after reducing the dimensions and tuning the baseline algorithms with parameters tuning would outperform the results achieved by the standalone baseline algorithms from previous work?**

The results for this research question is evaluated based on classifier metrics including weighted precision, weighted recall, f1-micro, and f1-macro. It makes sense to note the following notations along with how scikit-learn evaluate them below:



- y the set of *predicted* $(sample, label)$ pairs
- $\hat{y}$ the set of *true* $(sample, label)$ pairs
- L the set of labels
- S the set of samples
- y_s the subset of y with sample s, i.e. $y_s := \left\{(s', l) \in y | s' = s\right\}$
- y_l the subset of y with label l
- similarly, $\hat{y}_s$ and $\hat{y}_l$ are subsets of $\hat{y}$
- $P(A, B) := \frac{|A \cap B|}{|A|}$
- $R(A, B) := \frac{|A \cap B|}{|B|}$ (Conventions vary on handling $B = \emptyset$; this implementation uses $R(A, B) := 0$, and similar for P.)
- $F_\beta(A, B) := \left(1 + \beta^2\right) \frac{P(A,B) \times R(A,B)}{\beta^2 P(A,B) + R(A,B)}$

Figure 5: Notations

- Weighted Precision = $\frac{1}{\sum_{l \in L} |\hat{y}_l|} \sum_{l \in L} |\hat{y}_l| P(y_l, \hat{y}_l)$
- Weighted Recall = $\frac{1}{\sum_{l \in L} |\hat{y}_l|} \sum_{l \in L} |\hat{y}_l| R(y_l, \hat{y}_l)$
- F1-macro= $\frac{1}{|L|} \sum_{l \in L} |\hat{y}_l| F_\beta(y_l, \hat{y}_l)$
- F1-micro = $F_\beta(y, \hat{y})$

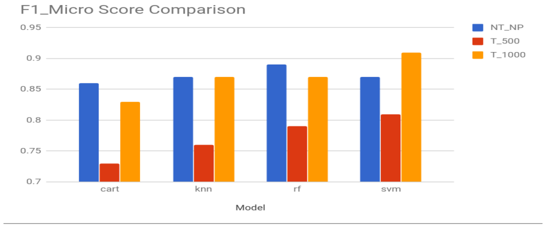


Figure 6: F1-Micro Score Comparison

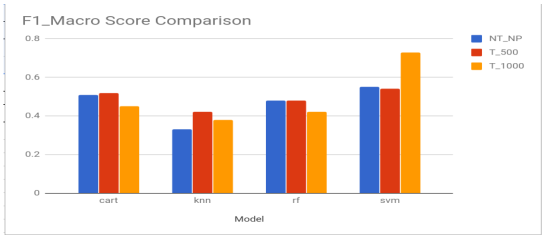
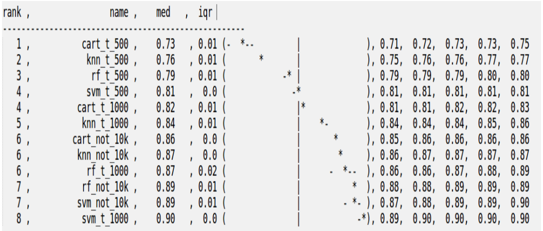


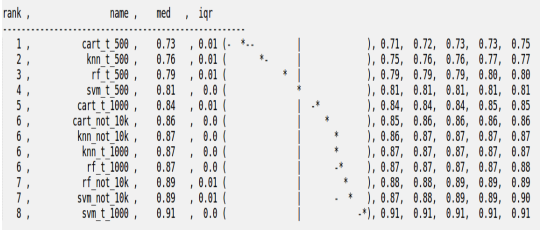
Figure 7: F1-Macro Score Comparison

The report for F1-Micro and F1-Macro are reported in the two graphs of Figure 6 and 7 above since the goal is to compare with the previous work by Athar. It is surprising that Athar did not use random forest in his work because the classifier result for it was higher from the graph. From the two graphs of F-micro and F-macro results, even when only 1000 dimensions (10% of the original dimension space 10,561 dimensions in Athar's paper [3]), tuning SVM performed better than the rest. However, are the result difference significant? Scott-Knott can be applied here to rank these results, shown in the few tables below (Figure 8-11).



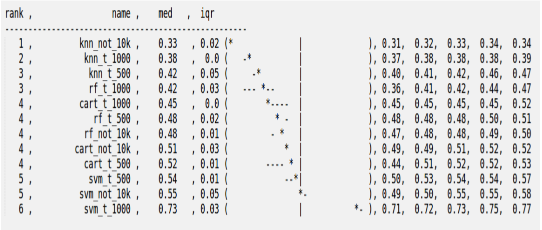
```
rank ,              name ,    med   , iqr
---------------------------------------------
   1 ,        cart_t_500 ,    0.73   , 0.01 (-  *--       |             ), 0.71,  0.72,  0.73,  0.73,  0.75
   2 ,         knn_t_500 ,    0.76   , 0.01 (      *      |             ), 0.75,  0.76,  0.76,  0.77,  0.77
   3 ,          rf_t_500 ,    0.79   , 0.01 (          -* |             ), 0.79,  0.79,  0.79,  0.80,  0.80
   4 ,         svm_t_500 ,    0.81   ,  0.0 (            -*             ), 0.81,  0.81,  0.81,  0.81,  0.81
   4 ,       cart_t_1000 ,    0.82   , 0.01 (             |*            ), 0.81,  0.81,  0.82,  0.82,  0.83
   5 ,        knn_t_1000 ,    0.84   , 0.01 (             |   *-        ), 0.84,  0.84,  0.84,  0.85,  0.86
   6 ,      cart_not_10k ,    0.86   ,  0.0 (             |      *      ), 0.85,  0.86,  0.86,  0.86,  0.86
   6 ,       knn_not_10k ,    0.87   ,  0.0 (             |       *     ), 0.86,  0.87,  0.87,  0.87,  0.87
   6 ,         rf_t_1000 ,    0.87   , 0.02 (             |     -  *--  ), 0.86,  0.86,  0.87,  0.88,  0.89
   7 ,        rf_not_10k ,    0.89   , 0.01 (             |         *   ), 0.88,  0.88,  0.89,  0.89,  0.89
   7 ,       svm_not_10k ,    0.89   , 0.01 (             |       - *-  ), 0.87,  0.88,  0.89,  0.89,  0.90
   8 ,        svm_t_1000 ,    0.90   ,  0.0 (             |           -*), 0.89,  0.90,  0.90,  0.90,  0.90
```

Figure 8: Precision



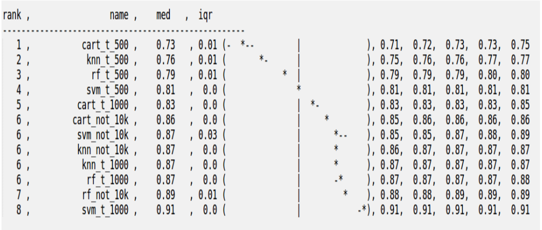
```
rank ,              name ,    med   , iqr
---------------------------------------------
   1 ,        cart_t_500 ,    0.73   , 0.01 (-  *--       |             ), 0.71,  0.72,  0.73,  0.73,  0.75
   2 ,         knn_t_500 ,    0.76   , 0.01 (      *-     |             ), 0.75,  0.76,  0.76,  0.77,  0.77
   3 ,          rf_t_500 ,    0.79   , 0.01 (          *  |             ), 0.79,  0.79,  0.79,  0.80,  0.80
   4 ,         svm_t_500 ,    0.81   ,  0.0 (             *             ), 0.81,  0.81,  0.81,  0.81,  0.81
   5 ,       cart_t_1000 ,    0.84   , 0.01 (             | -*          ), 0.84,  0.84,  0.84,  0.85,  0.85
   6 ,      cart_not_10k ,    0.86   ,  0.0 (             |    *        ), 0.85,  0.86,  0.86,  0.86,  0.86
   6 ,       knn_not_10k ,    0.87   ,  0.0 (             |      *      ), 0.86,  0.87,  0.87,  0.87,  0.87
   6 ,        knn_t_1000 ,    0.87   ,  0.0 (             |      *      ), 0.87,  0.87,  0.87,  0.87,  0.87
   6 ,         rf_t_1000 ,    0.87   ,  0.0 (             |      -*     ), 0.87,  0.87,  0.87,  0.87,  0.88
   7 ,        rf_not_10k ,    0.89   , 0.01 (             |        *    ), 0.88,  0.88,  0.89,  0.89,  0.89
   7 ,       svm_not_10k ,    0.89   , 0.01 (             |      -  *   ), 0.87,  0.88,  0.89,  0.89,  0.90
   8 ,        svm_t_1000 ,    0.91   ,  0.0 (             |           -*), 0.91,  0.91,  0.91,  0.91,  0.91
```

Figure 9: Recall



```
rank ,              name ,    med   , iqr
---------------------------------------------
   1 ,       knn_not_10k ,    0.33   , 0.02 (*            |             ), 0.31,  0.32,  0.33,  0.34,  0.34
   2 ,        knn_t_1000 ,    0.38   ,  0.0 (  -*         |             ), 0.37,  0.38,  0.38,  0.38,  0.39
   3 ,         knn_t_500 ,    0.42   , 0.05 (    -*       |             ), 0.40,  0.41,  0.42,  0.46,  0.47
   3 ,         rf_t_1000 ,    0.42   , 0.03 (  --- *--    |             ), 0.36,  0.41,  0.42,  0.44,  0.47
   4 ,       cart_t_1000 ,    0.45   ,  0.0 (       *---- |             ), 0.45,  0.45,  0.45,  0.45,  0.52
   4 ,          rf_t_500 ,    0.48   , 0.02 (         * - |             ), 0.48,  0.48,  0.48,  0.50,  0.51
   4 ,        rf_not_10k ,    0.48   , 0.01 (        - *  |             ), 0.47,  0.48,  0.48,  0.49,  0.50
   4 ,      cart_not_10k ,    0.51   , 0.03 (          *  |             ), 0.49,  0.49,  0.51,  0.52,  0.52
   4 ,        cart_t_500 ,    0.52   , 0.01 (       ---- *|             ), 0.44,  0.51,  0.52,  0.52,  0.53
   5 ,         svm_t_500 ,    0.54   , 0.01 (          --*|             ), 0.50,  0.53,  0.54,  0.54,  0.57
   5 ,       svm_not_10k ,    0.55   , 0.05 (             *-            ), 0.49,  0.50,  0.55,  0.55,  0.58
   6 ,        svm_t_1000 ,    0.73   , 0.03 (             |         *-  ), 0.71,  0.72,  0.73,  0.75,  0.77
```

Figure 10: F1-Macro



```
rank ,              name ,    med   , iqr
---------------------------------------------
   1 ,        cart_t_500 ,    0.73   , 0.01 (-  *--       |             ), 0.71,  0.72,  0.73,  0.73,  0.75
   2 ,         knn_t_500 ,    0.76   , 0.01 (      *-     |             ), 0.75,  0.76,  0.76,  0.77,  0.77
   3 ,          rf_t_500 ,    0.79   , 0.01 (          *  |             ), 0.79,  0.79,  0.79,  0.80,  0.80
   4 ,         svm_t_500 ,    0.81   ,  0.0 (             *             ), 0.81,  0.81,  0.81,  0.81,  0.81
   5 ,       cart_t_1000 ,    0.83   ,  0.0 (             | *-          ), 0.83,  0.83,  0.83,  0.83,  0.85
   6 ,      cart_not_10k ,    0.86   ,  0.0 (             |    *        ), 0.85,  0.86,  0.86,  0.86,  0.86
   6 ,       svm_not_10k ,    0.87   , 0.03 (             |      *--    ), 0.85,  0.85,  0.87,  0.88,  0.89
   6 ,       knn_not_10k ,    0.87   ,  0.0 (             |      *      ), 0.86,  0.87,  0.87,  0.87,  0.87
   6 ,        knn_t_1000 ,    0.87   ,  0.0 (             |      *      ), 0.87,  0.87,  0.87,  0.87,  0.87
   6 ,         rf_t_1000 ,    0.87   ,  0.0 (             |      -*     ), 0.87,  0.87,  0.87,  0.87,  0.88
   7 ,        rf_not_10k ,    0.89   , 0.01 (             |        *    ), 0.88,  0.88,  0.89,  0.89,  0.89
   8 ,        svm_t_1000 ,    0.91   ,  0.0 (             |           -*), 0.91,  0.91,  0.91,  0.91,  0.91
```

Figure 11: F1-Micro

Noting that the format is as <learner>_<not or t>_<number of dimensions>. Specifically, 'not' stands for not tuned model and 't' stands for tuned model. Looking across the four metrics, this study's model with 1000 dimensions perform significant better in SVM after tuning and only used 10% of the dimensionality space (especially in F1-Macro). For other learners, most no tuned and no reduced dimensionality ones have results that are more significant except kNN. For 500 dimensions, this study model performed worse than the one from the previous study for most of the metrics except for F1-Macro. It is safe to conclude that our model's performance is significantly better than the previous work by Athar.

***RQ 3:* What are the trade-off for applying SMOTE with self-tuning and are they acceptable?**

CPU time is one of the big trade-off for tuning since DE can take quite some time when encountering difficulty in converging even if they are faster than Genetic Algorithm. Specifically, it takes 300 times for running default Random Forest, 4 times more for CART, and almost twice for kNN. It is indeed faster for SVM which resulted in significant better performance discussed in ***RQ 2***. One thing that was unexpected was that the data quality of the model was very high which leads to lower variance and lesser noises so by applying SMOTE and tuning, the model's performance did not increase significantly. However, it is definitely in the case that this study only used 1000 dimensions in comparison with 10,561 dimensions from the original work. Therefore, by applying SMOTE and self-tuning the model with 10,561 dimensions, the results might have been different. Moreover, it would be difficult to know without the work with the imbalance nature of the dataset and sub-optimal default parameters performance nature. It is still safe to conclude that Athar for his work should have applying SMOTE and self-tuning.

5 THREATS TO VALIDITY

There are multiple threats to the validity aspect of the empirical results and associated answers that were made to research questions this study:

Selection of tuning criterias with model and tuning algorithms: model such as Fast-Frugal Tree and Naive-Bayes also have lower foot memory [15] beside CART, which are great candidates for learnability and repeatability criteria that were not considered for this study. Moreover, we only considered differential evolution algorithm for self-tuning criteria without looking into particle swarm optimization, alternative DE, and Bayesian optimization. Furthermore, tuning were not done on the number of dimensions to be reduced by PCA, the parameters of SMOTE, and other parameters in the learners.

Biases in the dataset: the dataset includes manually annotated sentiments. However, there were no report on the 310 papers were selected specifically (randomly or through text analysis) and how it was annotated (one person annotated the whole dataset, divide each out between multiple people, one citation context being annotated multiple times or one, etc). Moreover, there were no mention of how they validate if the annotated dataset have consistency and self-consistency. Consistency can be checked by having the citation context being annotated by two groups and apply t-test for statistically significance. Self-consistency can be checked with correlations between sets of labels and/or groups that doing the annotation. Furthermore, there were only 310 papers and specifically for the Computational Linguistic community, it is uncertain if that can represent the sentiment analysis well across other scholar community, e.g., Software Engineering and System (based on technical words and nature of the research in the community).

No pattern/local contexts: after attempting clustering the dataset first hopefully to find any local context or nature, it was found that there were no local context to be found (many clusters with one element, more than half of the dataset is in one cluster with the number of the cluster is more than 10, etc). However, with different ways of pre-processing the text or a different citation context dataset, the result might be different which leads to more interesting approach and solutions that can be found.

6 CONCLUSION

The collective goal of the study was to develop the learnability and self-tuning criteria of the tuned machine learning algorithms while decreasing the dimensions of space in the dataset and handling the imbalance class problem. Then, compare the model with the state of the art algorithms for sentiment analysis of scientific citations proposed in these previous work.

There is no best way to compare or rank the results of the baseline model with the tuned models, depending on the goals, the data, and the audience [14]. To find the best model for this sentiment analysis of scientific citation problem and dataset, it would be depending on the goal of the researcher and user which are repeatability and self-tuning for this study. In term of repeatability, by reducing the dimensionality and using random forest or CART, both storage space and time are much better with comparable results (random forest especially perform better). In term of self-tuning, our tuned SVM performed significantly better than the original SVM from Athar's work even when only 10% of dimensions were applied. However, looking into across all the learners, it did not outperform default Random Forest impressively. Moreover, by integrating both criteria, it is difficult to pick the one that satisfy both.

6.1 Lesson Learned

According to Fu's work [10], it is necessary to tune your machine learning algorithms instead of applying off-the-shelf default algorithms to your work. Moreover, with the imbalance class problem, modified SMOTE has shown potentials in improving data quality to significantly help with the results of the learners' performance [2]. Yet the classifier performance were not as impressive as expected. However, it is more important to do the investigation and handling the problem properly instead of making non-scientific assumptions because the result is unknown until it is investigated. Moreover, what works for other studies do not mean it will work for this specific project (tuning, clustering, ensemble methods, etc) due to many compounding factors. Data mining community's focus has been shifted to more of finding the best algorithm instead of improving the data quality. However, without data quality, model would not be able to perform well since it is data mining and algorithm mining. Finally, according to the No Free Lunch theorem, one cannot achieve everything or satisfy all the criteria.

6.2 Future Work

Transfer this work to do sentiment analysis of scientific citations in Software Engineering Conference and Journal Papers 35,391 Software Engineering papers from the last 25 years published in 34 top-ranked conferences and journals that have been used in [13] by:

(1) Sort out the relevant papers (250-350 papers)
(2) Preprocessing the dataset for citation contexts
(3) Manually label the authors, work, and citing work
(4) Manually label the sentiments for citations within those papers
(5) Validate the dataset (for class distribution, consistency, self-consistency)
(6) Dimensionality Reduction + Tuning

From the corpus of 35k Software Engineering papers and the network of citations (papers or authors or journals are presented by the nodes and the edges representing the referencing act), more opportunities can be found:

- disseminate and study problem, approach, model, etc within the Software Engineering community
- build a research work/model recommendation system for Software Engineering (SE) scholar community

REFERENCES

[1] Amjad Abu-Jbara, Jefferson Ezra, and Dragomir R. Radev. 2013. Purpose and Polarity of Citation: Towards NLP-based Bibliometrics. In *HLT-NAACL*.

[2] Amritanshu Agrawal and Tim Menzies. 2017. "Better Data" is Better than "Better Data Miners" (Benefits of Tuning SMOTE for Defect Prediction). *CoRR* abs/1705.03697 (2017).

[3] Awais Athar. 2011. Sentiment Analysis of Citations Using Sentence Structure-based Features. In *Proceedings of the ACL 2011 Student Session (HLT-SS '11)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 81–87. http://dl.acm.org/citation.cfm?id=2000976.2000991

[4] Awais Athar and Simone Teufel. 2012. Context-enhanced Citation Sentiment Detection. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT '12)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 597–601. http://dl.acm.org/citation.cfm?id=2382029.2382125

[5] James Bergstra and Yoshua Bengio. 2012. Random Search for Hyper-parameter Optimization. *J. Mach. Learn. Res.* 13 (Feb. 2012), 281–305. http://dl.acm.org/citation.cfm?id=2188385.2188395

[6] Luiz Coletta, Nadia Felix, Eduardo Hruschka, and Estevam Hruschka. 2014. Combining Classification and Clustering for Tweet Sentiment Analysis. (01 2014).

[7] Ying Ding, Guo Zhang, Tamy Chambers, Min Song, Xiaolong Wang, and Chengxiang Zhai. 2014. Content-based citation analysis: The next generation of citation analysis. *Journal of the Association for Information Science and Technology* 65, 9 (2014), 1820–1833. https://doi.org/10.1002/asi.23256

[8] Leo Egghe. 2007. Dynamic h-index: The Hirsch index in function of time. *Journal of the American Society for Information Science and Technology* 58, 3 (2007), 452–454. https://doi.org/10.1002/asi.20473

[9] Wei Fu and Tim Menzies. 2017. Easy over Hard: A Case Study on Deep Learning. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering (ESEC/FSE 2017)*. ACM, New York, NY, USA, 49–60. https://doi.org/10.1145/3106237.3106256

[10] Wei Fu, Tim Menzies, and Xipeng Shen. 2016. Tuning for Software Analytics. *Inf. Softw. Technol.* 76, C (Aug. 2016), 135–146. https://doi.org/10.1016/j.infsof.2016.04.017

[11] Daniel Gildea, Sanjeev Khudanpur, Anoop Sarkar, Simon Fraser U, Kenji Yamada, Alex Fraser, Libin Shen, David Smith, Johns Hopkins U, Katherine Eng, Stanford U, Viren Jain, Zhen Jin, and Dragomir Radev. 2004. A Smorgasbord of Features for Statistical Machine Translation. (05 2004).

[12] Angelo Di Iorio, Andrea G Nuzzolese, and Silvio Peroni. 2013. Towards the Automatic Identification of the Nature of Citations. *SePublica* (2013), 63âĂŞ74. http://ceur-ws.org/Vol-994/paper-06.pdf

[13] George Mathew, Amritanshu Agarwal, and Tim Menzies. 2016. Trends in Topics at SE Conferences (1993-2013). *CoRR* abs/1608.08100 (2016). arXiv:1608.08100 http://arxiv.org/abs/1608.08100

[14] Timothy Menzies. 2017. Fss17: Evaluation. (2017). https://txt.github.io/fss17/stats

[15] Nathaniel Phillips, Hansjoerg Neth, Wolfgang Gaissmaier, and Jan Woike. 2017. FFTrees: A toolbox to create, visualise, and implement fast-and-frugal decision trees. (2017). unpublished manuscript.

[16] Dragomir R. Radev, Mark Thomas Joseph, Bryan Gibson, and Pradeep Muthukrishnan. 2016. A bibliometric and network analysis of the field of computational linguistics. *Journal of the Association for Information Science and Technology* 67, 3 (2016), 683–706. https://doi.org/10.1002/asi.23394

[17] Matthias Scholz. 2006. *Approaches to analyse and interpret biological profile data*. doctoralthesis. Universität Potsdam.

[18] Jonathon Shlens. 2014. A Tutorial on Principal Component Analysis. *CoRR* abs/1404.1100 (2014). arXiv:1404.1100 http://arxiv.org/abs/1404.1100

[19] Ina Spiegel-RÃűsing. 1977. Science Studies: Bibliometric and Content Analysis. *Social Studies of Science* 7, 1 (1977), 97–113. http://www.jstor.org/stable/284635

[20] Simone Teufel, Advaith Siddharthan, and Dan Tidhar. 2006. Automatic Classification of Citation Function. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP '06)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 103–110. http://dl.acm.org/citation.cfm?id=1610075.1610091