UNIVERSITY POLITEHNICA OF BUCHAREST
FACULTY OF AUTOMATIC CONTROL AND COMPUTERS
COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

DIPLOMA PROJECT

Anomaly Detection on Scientific Publications

David-Gabriel ION

Thesis advisor:

Prof. dr. ing. Ciprian DOBRE

BUCHAREST

2022

CONTENTS

ABSTRACT

This work proposes a method for automatically determining when an author publishes a scientific paper in a field outside his expertise. The system can solve the situation when there are two authors with the same name and a publication whose author is one of those two people, offering a score proportional with the probability that each author wrote the aforementioned publication. Using machine learning algorithms, it is possible to determine both the topics associated with a publication and the fields an author is proficient in. Using this user profile, it is possible to estimate how likely it is that the author wrote the newly analysed publication. The current approach uses algorithms train on only the provided dataset, the optimal parameters being chosen manually. As a result, the system has an accuracy of approximately 81%. The shortcomings of this approach are the reduced dataset, the system classifying usual words as technical terms, but also the limited flexibility of the chosen algorithms. The system can be optimized by using machine learning models pretrained on large datasets and end-to-end training the whole model.

1 INTRODUCTION

This project came to be when the need for an automated importing of publications mechanism was needed for the CRESCDI Research website. When a publication is imported, it is desirable to have it link to the user who wrote it. In order to solve this issue, an anomaly detection algorithm was proposed, since having an author publish an article in a domain outside their area of expertise would clearly be flagged as an anomaly. While this project does not specifically focus on anomaly detection per se, since there are sufficient positive and negative examples in the dataset, it borrows the intuition from it.

One main assumption which was made is the fact that publications can be classified as belonging to a topic, and users generally publish articles in their topic or topics of expertise. While this is a fairly sane assumption, as the vast majority of authors only occasionally stray from their general field, the implementation may need to take into account several complex aspects, such as the relations between the topics and the appearance or disappearance of certain topics (such as block chain and outperformed algorithms). In the presented solutions, we will both ignore and treat this aspect and compare the results, along with several other aspects presented later.

1.1 The Problem

The main problem that this project aims to solve is the name ambiguity caused by the fact that there can be multiple different authors with the same name. When a new publication is added to a database, it is desirable to have its authors associated with the publication. This is sometimes difficult, since the publications can come from different sources and having different formats.

1.2 Objectives

The scope of the project is to provide a numerical score proportional to the probability that a given author wrote a given publication. Another objective is to update the model automatically as new publications are added, thus removing the need to periodically retrain the entire system.

1.3 Proposed Solution

The proposed solution is to extract the author's name from a new publication and test the publication against all authors in the database with the extracted name. If the model assigns a score larger than a threshold, then the association between the publication and the author is marked as likely, otherwise it is marked as unlikely. All associations will pass through a manual review phase, where either the user or a site administrator can decide whether the association is correct or not, and they can see the model's predicted score. If the association is not reviewed in a given time, then the decision falls to the system. As a result, the system will not make any unsupervised decision without the opportunity for human intervention and will mostly aid in sorting the associations by likelihood and graphically augmenting the user interface.

In order to actually provide the aforementioned score, the system will be trained on the CRESCDI Research dataset. The publications' abstracts will be inputted into a topic modelling algorithm, which will determine the associated topics for each publication. In other words, each publication will have a feature vector associated, where each entry in the vector represents how closely the publication matches each topic. With this, we can construct the user profile of each user, which indicates the topics they are proficient in. The score is given by the similarity between the newly added publication and the user profile.

1.4 Results

So far, the accuracy of the system is approximately 81%. This result is while not using pretrained models or end-to-end training, meaning it can be drastically improved with those two enhancements alone. Even as-is, it can serve as a tool to ease the manual author approval job.

1.5 Structure

The following sections present the general proposed solution, which allows for various topic models to be used. Those considered topic models are presented below, along with their known issues and expected results, but also how they performed when applied on this problem. Next, the actual training method is described, how the model fine tuned and what performance it ultimately has.

2 PROPOSED SOLUTION

The general proposed solution follows the following steps:

1. Preprocess the dataset
2. Train the topic modelling algorithm on the training dataset
3. Compute the feature vector for each publication in the training dataset
4. Compute the user profile for each author using the feature vectors associated with publications the authors wrote
5. Compute the feature vector for each publication in the cross-validation dataset
6. Evaluate the model's performance on the cross-validation dataset
7. Assess the results and if improvements are believed to be possible, go back to step 2.
8. Compute the feature vector for each publication in the test dataset
9. Evaluate the model's performance on the test dataset

The motivation for this general process is the easy integration with existing topic modelling algorithms. While alternatives, such as directly computing the user profile, would have required more complex and custom solutions, this approach relies on the accuracy and properties of the generated feature vectors.

If the topic modelling algorithm computes feature vectors closely related for publications in similar fields, then it is easy to compare the similarity between two publications. However, in our case, we need to compare the similarity between a user profile and a publication. As a result, we can represent the user profile as another feature vector, computed as the mean of feature vectors associated with the authored publications. The actual comparison also takes into account the standard deviation of the considered feature vectors, to account for authors

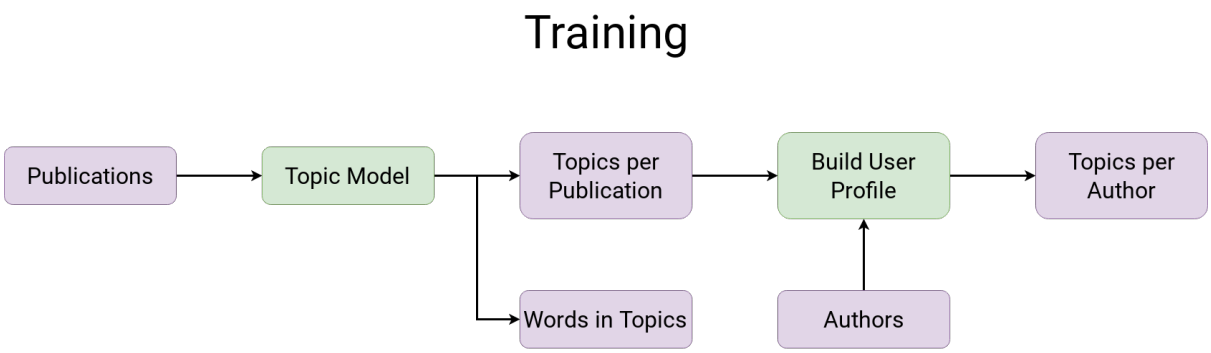

Figure 1: General Solution Training diagram. This describes the initial training step, which generates the topics' feature vectors. The topic model can be validated by the words associated with each topic.

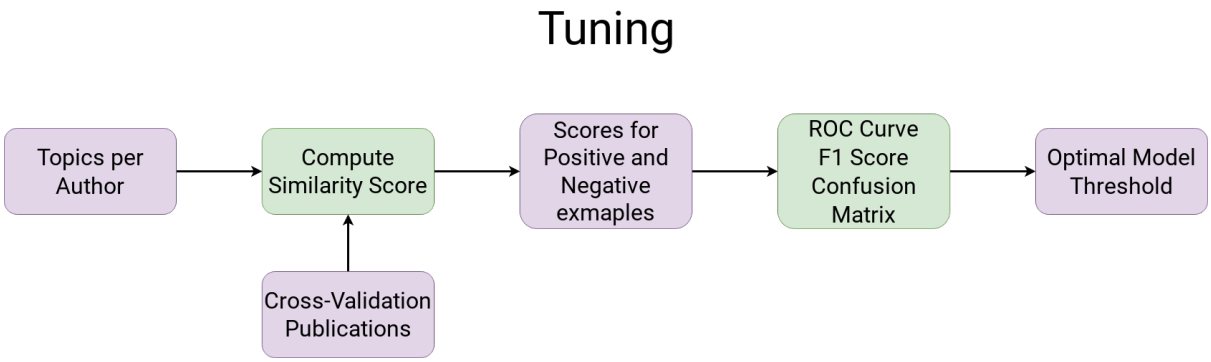

Figure 2: General Solution Tuning diagram. The purpose of this step is to find the optimal threshold, which separates the positive examples from the negative examples.

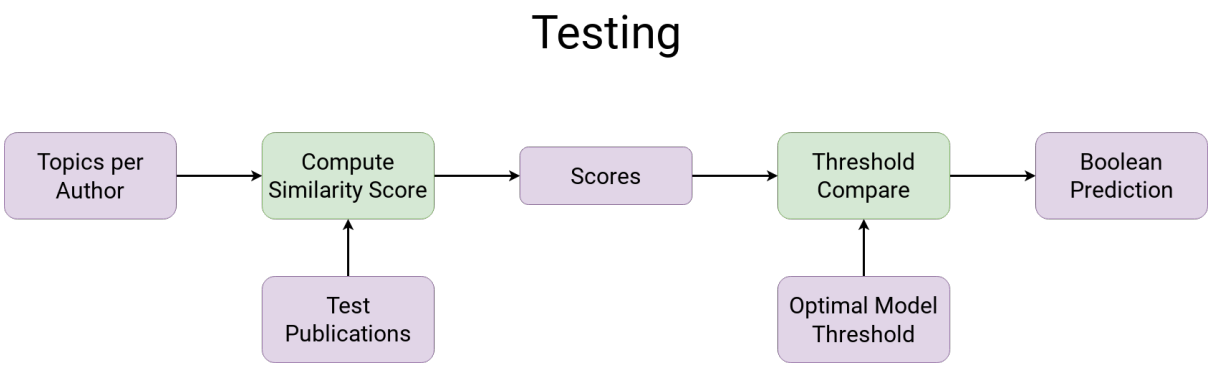

Figure 3: General Solution Testing diagram. Yields the expected performance on new data.

with varied areas of expertise.

Another reason for this approach is the easy analysis of the topic modelling algorithm. The existing algorithms all come with some way of manually inspecting the generated topics. This is especially useful in identifying the causes for poor performance.

Finally, this approach allows for online learning the system, as long as the topic modelling algorithm supports it as well. The general strategy would be to update the topic modelling algorithm and periodically update the user profiles as well. It is possible to update the mean and standard deviation of a user profile by removing the old feature vector and adding the new feature vector, without iterating over all the authored publications. As long as the topic modelling algorithm does not output drastically different data, this method allows for a rolling update over the entire dataset.

2.1 Latent Dirichlet Allocation

The general proposed solution does not imply the use of a particular topic model. In the initial implementation, Latent Dirichlet Allocation (LDA) was used. The basic principle of LDA is generating a model based on the correct topic allocation. After the model is defined, its parameters are optimized, which ultimately serve as the output of the model. This is the case

because the model is defined in such a way that topics are required for document generation.

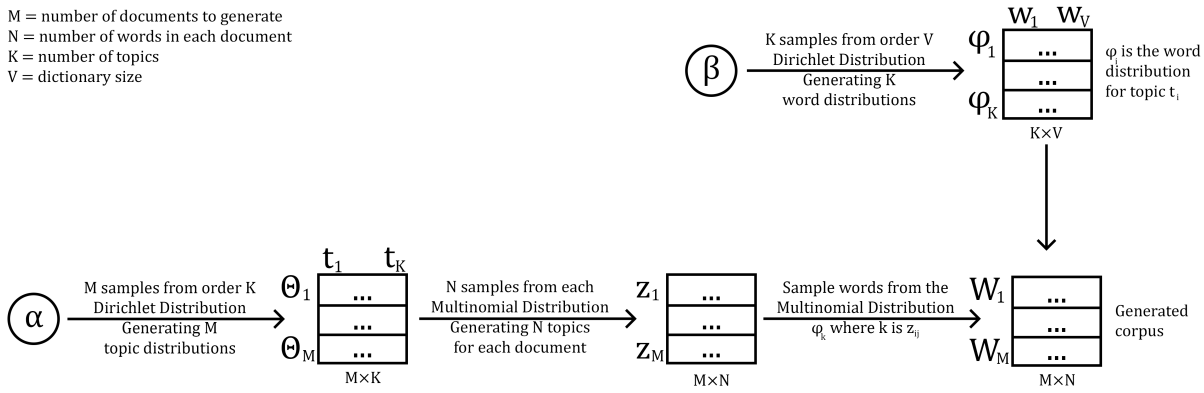

Figure 4: LDA Corpus Generator Architecture

The LDA model is trained by optimizing the two variables α and β. α represents the Dirichlet distribution from which topic distributions are chosen. When generating a corpus, this distribution is sampled M times, where M is the number of documents that we want to generate in the corpus. Since we are sampling from a K-order Dirichlet distribution, the result is an $M \times K$, where K is the number of topics the model uses. A Dirichlet distribution was used here because of its properties to chose vectors that have one dominant component. This matches the intuition that generally a document mostly belongs to a single topic. This means that the resulted matrix θ is mostly sparse, or, intuitively, we chose a single topic for each document.

Once the topic distributions are chosen, each distribution is sampled N times, where N is the number of words in each document. In practice, separate document lengths N_i chosen from Poisson distributions are used, but the implementation does not greatly differ. The $M \cdot N$ samples result in an $M \times N$ matrix Z of topics, where each value represents the topic associated with each word in the final corpus.

Independently, we can compute the word distributions for each topic, which will be used to sample the words of the generated corpus. These distributions are vectors sampled from the V-order Dirichlet distribution β, where V is the size of the vocabulary. Again, a Dirichlet distribution is used here for its properties to produce sparse results. In this case, each topic will have associated only a few words, so we can intuitively think that this selects the top few words for each topic. The word distributions are defined as ϕ

After both Z and ϕ are computed, we can iterate over all the elements of Z, z_{ij} and sample the multinomial distribution $\phi_{z_{ij}}$ for the word w_{ij}. Once the entire matrix Z is sampled, the resulted matrix W represents the generated corpus. A model has a higher accuracy if the probability of generating a corpus equal to the inputted corpus is as high as possible. Analytically, the function that we want to optimize is:

$$P(W, Z, \theta, \phi; \alpha, \beta) = \prod_{i=1}^{K} P(\phi_i; \beta) \prod_{j=1}^{M} P(\theta_i; \alpha) \prod_{t=1}^{N} P(Z_{jt}|\theta_j) P(W_{jt}|\phi_{Z_{jt}})$$

There are several ways to optimize this function. The original paper used a Monte Carlo simulation and Gibbs sampling was later suggested. The method used by this work follows the implementation described in *Online Learning for Latent Dirichlet Allocation*, which allows for online learning the model.

The main shortcoming of this topic model is the fact that there are no pretrained models already available, so in order to successfully differentiate between common words and technical terms, a large dataset is needed. The training dataset used for this project contains about 20.000 documents, which proved insufficient, due to the fact that words such as *two* or *results* were marked as topic keywords.

2.2 BERT

An implementation which solves the issue of insufficient data in the dataset is incorporating a pretrained transformer, which, intuitively, will behave much better even on a small dataset. This would also fix several other issues that the previous method had, such as word ambiguity (when the same word has different meanings depending on the context), long train times (BERT only needs fine tuning), limited performance (systems using BERT have shown better results than traditional algorithms for the same problems) and difficulty in training (since BERT is already trained, we can expect a reasonable performance out of the box).

This method is not yet implemented, but will be in a future iteration of the project.

3 PERFORMANCE TUNING

In order to train the system, the publications dataset was split in the training set (60% of the data), the cross-validation set (20% of the data) and the test set (20% of the data). After training on the training set, the performance was analysed on the cross-validation set. Because the scores outputted by the system were arbitrary, a threshold value had to be chosen to separate the negative and positive predictions.

To do this, the ROC curve was computed. When using the negative examples in the dataset, the graph was indicating a poor model performance. Part of the reason for this was the imbalance in the dataset. The ration between positive and negative examples is about 7:1, which makes it more difficult to obtain optimal performance. Fixing this issue would have been possible by using the FBeta score, but that introduces an extra hyperparameter.

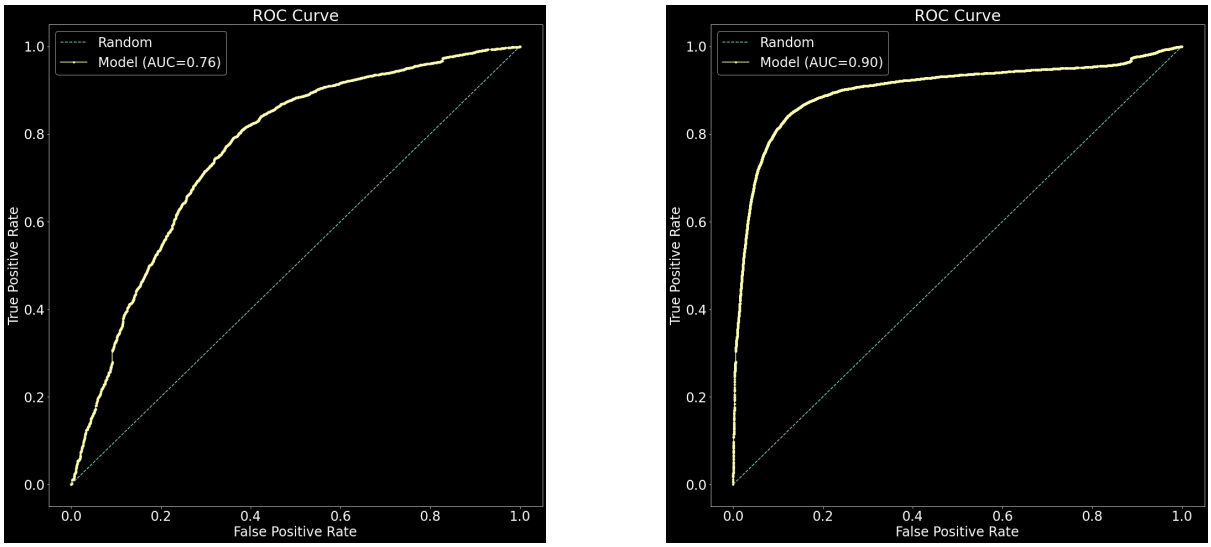

Figure 5: ROC Curves on the original dataset (left) and on the dataset with synthetic negative examples (right)

Working around this issue was the generation of synthetic negative examples, by randomly choosing publication and author pairs until a dataset as large as the positive examples dataset was formed. The reasoning for creating this dataset was the small probability of choosing positive examples. As a result, the vast majority of the examples would be negative. Afterwards, the new ROC curve indicated better performance.

In order to set a threshold, the objective was chosen as maximizing the F1 score. This decision was because the F1 score offered a great balance between false positives and false negatives. Instead of computing the F1 score for each possible threshold, the ROC curve was sampled at regular intervals, and only those sample points were considered as potential thresholds.

The reasoning behind this is that the thresholds may vary greatly before showing different results, and rigorously optimizing the F1 score would not offer great performance benefits. As a result, the ROC curve was sampled at 0.01 increments for the false positives value in a reasonable interval (0.01 to 0.4). For each sample point, there is an associated threshold. After this, the threshold associated with the highest F1 score among the samples is chosen. This concluded the training and the model is now ready to make predictions on new data.

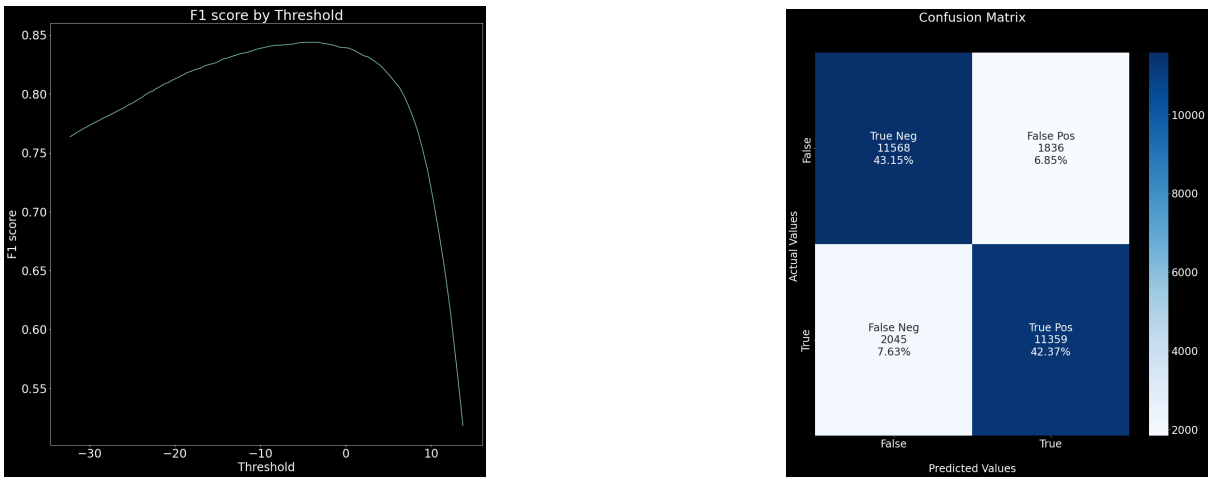

Figure 6: F1 score by threshold (left) and the generated confusion matrix (right) when using synthetic negative examples. The accuracy on this hybrid dataset was 86%.

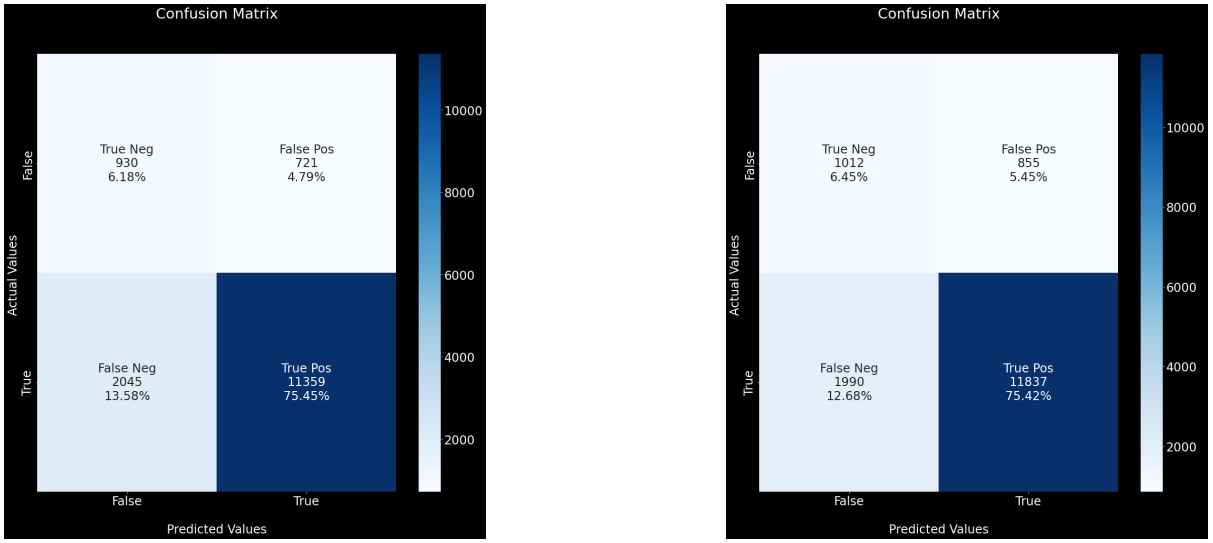

Figure 7: Confusion matrices generated on the cross-validation set (left) and on the test set (right). The accuracy in both cases was 81% and the F1 scores were 0.84 on the cross-validation set and 0.89 on the test set.

4 CONCLUSIONS

Overall, the final F1 score for the system was 0.89 and the accuracy 81%. When taking into account the use-case in which this system will be used, it would be highly desirable to increase the performance. There is room for improvement by switching to a topic model which incorporates a pretrained transformer, such as BERT or one of its variations. Also, incorporating the statistical anomaly detection algorithm into an end-to-end trainable model would greatly increase the performance, since it would allow the transformer to fine-tune more efficiently.