Book Genre Classification using RoBERTa and TFIDF with different classifiers

Text Mining Project
(732A81)

Author:
Abhijeet Anand
(abhan872)

Abstract

Book genre classification is the task of assigning a genre label to a given book based on its content. In this project, we explore the use of the RoBERTa language model and TF-IDF(Term Frequency-Inverse Document Frequency) as the feature extractors and several classifiers, including XG Boost Classifier, Multi-layer perceptron (MLP) classifier, support vector classifier (SVC), and others, are taken to perform the book genre classification. These classifiers are hyper-tuned using the grid search method to choose the best hyperparameters for the task of classification. We use a dataset of books with 6 different genres and evaluate the performance of our approach using various metrics. Our experimental results show that with TF-IDF as the feature extractor and MLP as the classifier, we achieved a test accuracy of 76.8% while using RoBERTa as a feature extractor and the SVC as the classifier we achieved a test accuracy of 68.7%.

1 Introduction

Book genre classification is the task of predicting the genre of a given book based on its content. This can be useful for a variety of practical applications, such as book recommendations to readers, organizing books in a library, online bookstores, or publication companies. In recent years, machine learning techniques is widely used for the classification of books into several genres.

One such approach is to use a language model like RoBERTa (Liu et al., 2019) to extract features from the title and summary of a book. RoBERTa is a transformer model, which got pre-trained on raw texts from a huge corpus of English data. In general, the use of RoBERTa and other machine learning techniques has shown great results in similar classification tasks, and others(Liu et al., 2019). These approaches may also have the potential to improve the organization and recommendation of books in various settings.

In this project, RoBERTa is used as a feature extractor for the given data. The extracted features are then fed to the classifiers, which further learn to predict the genre of a book based on the extracted features by RoBERTa.

On the other hand, TF-IDF which is a very common technique in natural language processing and information retrieval tasks is also used to extract information from the data. This is followed by feeding this information to all the classifiers to learn and predict the genre of the given book.

2 Theory

2.1 RoBERTa

RoBERTa (Robustly Optimized BERT Approach) is a transformer-based language model developed by researchers at Facebook AI in 2019 (Liu et al., 2019). It is one of the variants of the popular BERT (Bidirectional Encoder Representations from Transformers) model. This model was designed to improve upon the original model by addressing some of its limitations.

The key difference between RoBERTa and BERT is that RoBERTa was trained on a much larger dataset. This includes a combination of both books and web pages (Liu et al., 2019). Training on such a large dataset allows RoBERTa to improve its performance on a variety of tasks.

Another difference is in their training process. RoBERTa uses a dynamic masking strategy which allows the model to predict a masked token based on the context of the surrounding tokens, rather than the entire sequence (Liu et al., 2019). This approach is more effective at improving the model's generalization abilities (Liu et al., 2019).

RoBERTa has achieved state-of-the-art results on several benchmarks, including the GLUE and SQuAD datasets (Liu et al., 2019). It has also been used to improve performance on a variety of natural language processing tasks, such as machine translation, text classification, and question answering (Liu et al., 2019).

Overall, RoBERTa is a powerful language model that has shown promise in a wide range of NLP tasks.

2.2 TF-IDF

TF-IDF (Term Frequency-Inverse Document Frequency) is a statistical measure that is widely used in text mining and natural language processing tasks as a way to represent the importance of a word in a document. The measure is composed of two parts: term frequency (TF) and inverse document frequency (IDF). TF measures the number of times a term (word) appears in a document, while IDF measures the rarity of the term in the entire corpus of documents. The idea behind using TF-IDF is that a term that appears frequently in a document, but not in many documents throughout the corpus, is likely to be more informative and relevant to the topic of the document than a term that appears frequently across all documents. The TF-IDF measure was first proposed by Karen Spärck

Jones in her 1972 paper "A statistical interpretation of term specificity and its application in retrieval" (Sparck Jones, 1972). Since then it has been widely used in various text mining and information retrieval tasks such as document retrieval, text classification, and feature extraction.

2.3 Classifiers

Classifiers can be defined as an algorithm, which is able to decide the categories of the data on the basis of the learned features. The motive of a classier is to assign a label to an unseen data based on the pattern learned through seen/training data. In this project, the results from 6 classifiers are compared. XG-Boost, multi-layer perceptron (MLP), support vector classifier (SVC), stochastic gradient descent (SGD), Random Forest, and Adaboost are chosen for this book genre classification task. The choice of these classifiers is done through considering several aspects like flexibility, ease of implementation, computational resources, and data size. Some benefits and restrictions of these classifiers are mentioned in their respective sections below.

2.3.1 Multi-Layer Perceptron Classifier

A multi-layer perceptron (MLP) is a type of artificial neural network that is composed of multiple layers of interconnected nodes, or "neurons." The MLP is a feedforward network, which means that information flows through the network in only one direction, from input to output.

MLPs are a common choice for classification tasks, where the goal is to predict a class label for the given input. The input layer gets the input data, and each subsequent layer processes and transforms the data using a set of weights and biases. The output layer produces the final prediction based on the transformed data.

MLPs have been widely used in various applications, including image classification, natural language processing, and speech recognition (Bengio et al., 1994). MLPs are capable of handling large and diverse datasets but may require good computational resources as well. They are generally considered to be powerful models for learning non-linear relationships in data.

2.3.2 XG Boost Classifier

XG Boost (eXtreme Gradient Boosting) is a decision tree-based ensemble machine learning algorithm that is widely used for classification and regression tasks (Chen and Guestrin, 2016). It is an efficient and accurate algorithm that has been commonly used in the industry and has achieved state-of-the-art results on many benchmarks (Chen and Guestrin, 2016).

One of the key features of XG Boost is its ability to handle large and imbalanced datasets, as well as its ability to handle missing values (Chen and Guestrin, 2016). It also has several tuning parameters that allow the user to customize the behavior of the model, such as the learning rate and the maximum depth of the trees (Chen and Guestrin, 2016). XG-Boots performs effectively when the data contains a good number of features and detected patterns are non-linear.

Overall, XG Boost is a powerful and popular machine-learning algorithm that is well-suited for a variety of tasks, including classification and regression.

2.3.3 Random Forest

Random forest is another machine learning algorithm that constructs a set of decision trees and combines their predictions to classify or predict a target variable (Breiman, 2001). It is a popular method for both classification and regression tasks and has been used in a wide range of applications, including image and speech recognition, natural language processing, and bioinformatics ((Breiman, 2001); (Liaw and Wiener, 2001)).

Random forest works by training multiple decision trees on different subsets of the training data and uses the majority vote of the trees to make a final prediction (Breiman, 2001). This algorithm helps to reduce over-fitting and improve the generalization performance of the model.

Random forest has several advantages, including its ability to handle large datasets, high dimensionality, and missing values ((Breiman, 2001); (Liaw and Wiener, 2001). It is relatively easy to implement as it can produce good results with minimal parameter tuning (Breiman, 2001).

2.3.4 Stochastic Gradient Descent

Stochastic Gradient Descent (SGD) is a widely-used optimization algorithm for training machine learning models. This algorithm is very commonly used in deep learning. It is a variant of gradient descent where the model's parameters are updated incrementally with each training example. This makes SGD more computationally efficient, especially when working with large datasets.

The original paper on SGD was published in

1951 by Robbins and Monro (Robbins and Monro, 1951) and it has been widely used and improved upon since then.

2.3.5 Adaboost

AdaBoost (Adaptive Boosting) is a popular ensemble learning algorithm that is used to improve the performance of weak classifiers. Adaboost combines multiple weak learners to form a stronger classifier. The algorithm was first introduced by Yoav Freund and Robert Schapire in their 1996 paper "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting" (Freund and Schapire, 1997).

Adaboost is an iterative algorithm that adjusts the weights of the training examples at each iteration based on the error of the previous iteration's classifier. It continues to add classifiers until a certain accuracy or a maximum number of classifiers is reached. Adaboost is commonly used in a variety of applications such as image and speech recognition, computer vision, and bioinformatics.

Adaboost is known to generate good results with imbalanced datasets and or when the classes are difficult to separate.

2.3.6 Support Vector Classifier

Support Vector Classifiers (SVCs) are a type of supervised machine learning algorithm that is used for classification tasks. The idea behind SVCs is to find a hyperplane that can separate the different classes in the feature space. The boundary is chosen such that it maximizes the margin, which is the distance between the boundary and the closest data points from each class. The data points that are closest to the boundary are called support vectors.

The algorithm was first introduced by Vladimir Vapnik in his 1995 paper "The Nature of Statistical Learning Theory" (Vapnik, 1995). SVCs have been widely used in various fields such as text and image classification, bioinformatics, and financial analysis.

SVCs are commonly used with a kernel trick, which allows them to handle non-linearly separable data, by mapping the data into a higher dimensional space.

3 Data

The data used in this project is taken from Kaggle. It initially has 4 columns namely index, title, genre, and summary with 4657 rows. The data contains 10 genres namely, thriller, fantasy, science, history, horror, crime, romance, psychology, sports, and, travel. This dataset is significantly imbalanced which is demonstrated in Fig. 1.

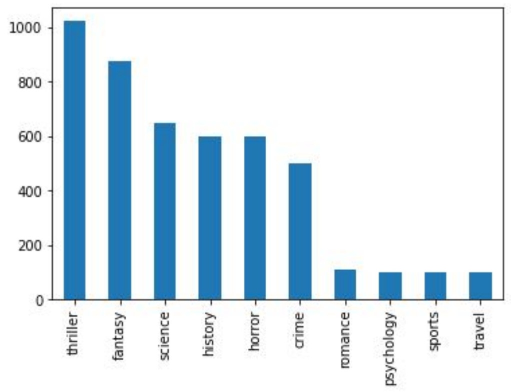


Figure 1: Number of books per genre

The word count in the summary field is observed to vary as well which ranges from 1 word to 5663 words as shown in Fig 2.

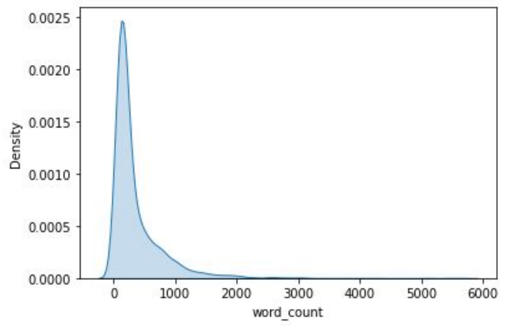


Figure 2: Density plot based on word count

Few of the books in the dataset contain the same name, and summary, but are labeled to different genres. This might confuse our model during the training.

During the basic cleaning, the column index and duplicated items were removed. Further, genres with less than 150 books and books with summary less than 100 words or more than 1500 words are removed. Finally, a dataset with 3102 rows and 6 genres (fantasy, science, crime, history, horror, and, thriller) was kept for preprocessing and modeling.

4 Methods

With the dataset of 3102 rows, two interesting columns title and summary are considered for feature extraction.

RoBERTa is a complex method for text representation or feature extraction. It is based on a deep neural network architecture, which is able to learn representations through capturing the semantics and syntax of the language. Therefore, it doesn't depend much upon pre-processing like other traditional feature-based methods. In few cases, basic pre-processing like lowercasing the text input might be helpful but not necessary.

To use TF-IDF, the data needs to be converted into bag of words, where every document represents a vector containing word frequencies. Therefore, TF-IDF generally needs pre-processing steps which involves tokenization, removal of stopwords, and, lemmatization/stemming. In this project, the mentioned preprocessing was divided into two steps basic cleaning, and data preprocess.

4.0.1 Basic cleaning

In the basic cleaning, all the data present in column Title and Summary were made lower case followed by removing punctuations, and, stopwords. Further, words with a length of less than 3 were also eliminated. Finally, single characters, HTML tags, new line characters, and multiple spaces were also removed. Most of these tasks were done with the help of regex library.

4.0.2 Data Pre-processing

In this section, the previously cleaned data is taken to further process with advanced processings like tokenization and lemmatization. At first, the columns, title, and summary are combined to make a new column 'title_summary'. The abovementioned processing is done on this new column and is finally exported to prepare document representation using TF-IDF.

4.1 Document Representation

An important step in any NLP task is document representation, where the text data is converted to number representation which can be used as input to machine learning models. There are several techniques to do this task including word embeddings, doc2Vec, transformers and others. In this project, RoBERTa (a transformer model) and TF-IDF are used.

In order to generate the document representation using RoBERTa model, the raw data including title and summary is fed to the processor which generates the input sequence with the help of tensflow_hub. This input sequence is then fed to encoder to get the output. The 'sequence_output' from the outputs has the shape of ([3102, 128, 512]). This is iterated over the range of length of data to get the final document representation using RoBERTa. In this project, a pre-trained RoBERTa model is used to generate the document representation.

To retrieve the information using TF-IDF, the pre-processed data is taken and fed to the 'TfidfVectorizer' from the 'sklearn' library is used. The generated document has a shape of (3102, 42306), where 3102 signifies the number of documents in the corpus and 42306 represents the number of unique words in the corpus of the given data.

4.2 Modelling

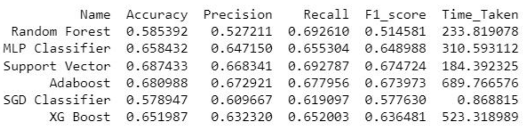
Modeling using both the document representations, RoBERTa and TF-IDF is performed. The extracted features are divided into train and test with a ratio of 70/30. All the classifiers used in this project were hyper-tuned with the help of the Grid Search Method. The results from all the classifiers along with the time taken are stored in a data frame and will be discussed later in this report.

5 Results

With the hyper-tuned parameters, all the classifiers were run on both document representations and the results were collected.

5.1 Results from RoBERTa model

As shown in Fig. 3, all the classifiers produce similar results. However, with an accuracy of 68.7% and precision of 66.8%, the Support Vector Classifier tops the table. Other metrics like Recall and F1-score are also recorded and can be referred to in Fig. 3.



```
           Name  Accuracy  Precision    Recall  F1_score  Time_Taken
  Random Forest  0.585392   0.527211  0.692610  0.514581  233.819078
 MLP Classifier  0.658432   0.647150  0.655304  0.648988  310.593112
 Support Vector  0.687433   0.668341  0.692787  0.674724  184.392325
       Adaboost  0.680988   0.672921  0.677956  0.673973  689.766576
 SGD Classifier  0.578947   0.609667  0.619097  0.577630    0.868815
       XG Boost  0.651987   0.632320  0.652003  0.636481  523.318989
```

Figure 3: Results from all classifiers with RoBERTa

The bar chart for accuracy concerning different classifiers can be seen in Fig. 4. Here, we see that all the classifiers have similar accuracy in the classification. With SVC having the highest accuracy of 68.7%, we also observe that SGD Classifier has the least accuracy of 57.8%.

The entire modelling was done over the CPU. So, the time consumption is more than usual. While the accuracy were very close, the time taken by each

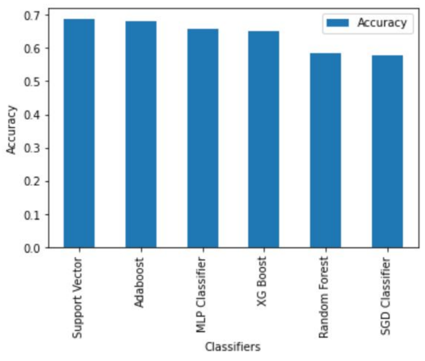


Figure 4: Accuracy from all classifiers

classifier varied significantly. Fig. 5 shows the time taken by different classifiers with RoBERTa model. It is observed that there is a big gap in time consumption. With Adaboost taking the maximum time of 689.7 seconds, SGD is observed to consume just 0.86 seconds.

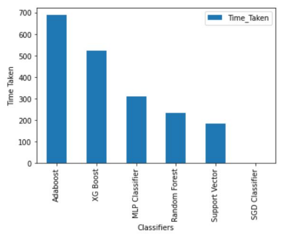


Figure 5: Time taken by classifiers

The confusion matrix is a useful tool for evaluating the performance of a classification model, and it can be easily computed using a variety of software tools and libraries. Looking at the confusion matrix from the best classifier, ie. SVC is shown in Fig. 6, it can be noticed that genre ‘Fantasy‘ is wrongly predicted as Thriller in vast.

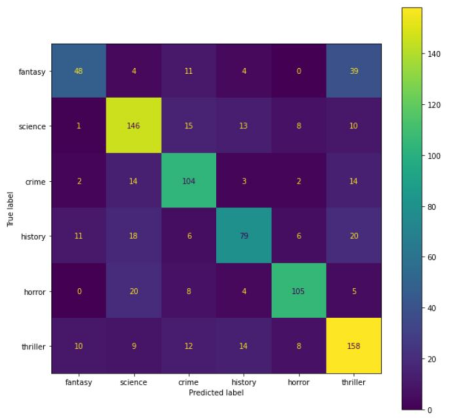
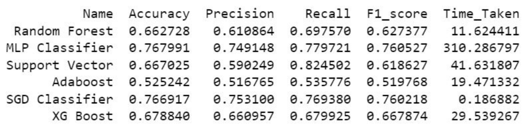


Figure 6: Confusion Matrix using RoBERTa

5.2 Results using TF-IDF

The results from all the classifiers using TF-IDF are summarized in 7. The maximum test accuracy of 76.8% is achieved using MLP Classifier, while SGD Classifier produced almost the same results with an accuracy of 76.7%. SGD achieves better precision of 75.3%, while MLP is very close to the precision value of 74.9%. Other metrics like Recall and F1-score are also recorded and can be referred to in Fig. 7.



```
             Name  Accuracy  Precision    Recall  F1_score  Time_Taken
    Random Forest  0.662728   0.610864  0.697570  0.627377   11.624411
   MLP Classifier  0.767991   0.749148  0.779721  0.760527  310.286797
   Support Vector  0.667025   0.590249  0.824502  0.618627   41.631807
         Adaboost  0.525242   0.516765  0.535776  0.519768   19.471332
   SGD Classifier  0.766917   0.753100  0.769380  0.760218    0.186882
         XG Boost  0.678840   0.660957  0.679925  0.667874   29.539267
```

Figure 7: Results from all classifiers with TF-IDF

The bar chart for accuracy concerning different classifiers using TF-IDF can be seen in Fig. 8. Here, it can be observed that MLP and SGD classifiers perform sigficantly better than any other classifier. It is also noticed that has the least accuracy of 52.5%.

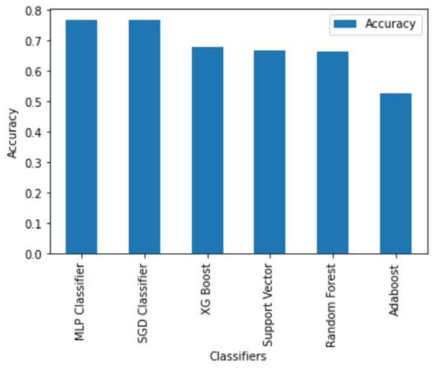


Figure 8: Accuracy from all classifiers

While using TF-IDF, the time taken by the MLP classifier was much greater than any other classifier. Fig. 9 shows the time taken by different classifiers with TF-IDF. With MLP taking the maximum time of 310.2 seconds, SGD is observed again to consume just 0.18 seconds.

The confusion matrix from the best classifier, MLP is shown in Fig. 10. In this case, as well, it can be noticed that the genre ‘Fantasy‘ is wrongly predicted as Thriller in vast.

Discussion

The confusion matrix from both the document representations had shown that the significant number

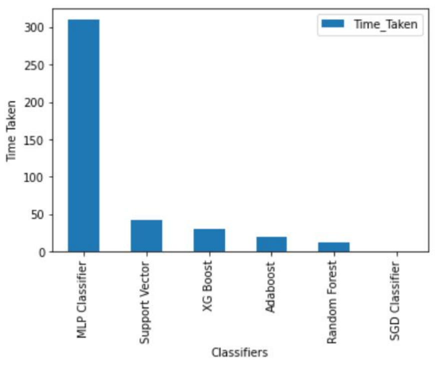


Figure 9: Time taken by classifiers

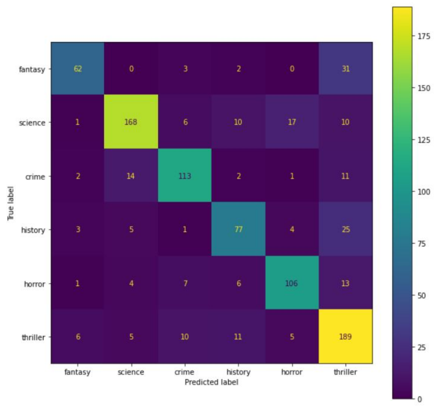


Figure 10: Confusion Matrix using TF-IDF

of fantasy books were predicted as thriller. Upon investigation for both the classes through word cloud, shown in Fig. 11 and Fig. 12, it was observed that both the genres are very closely related. Words like ‘find, life, time, world‘ are very common in both genres. Thus, the mis-classification between these two genres occur in vast.

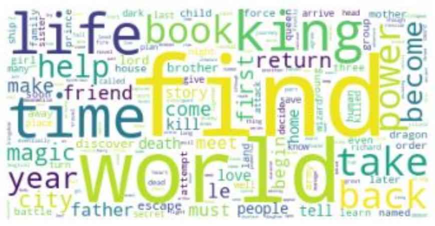


Figure 11: Word Cloud for genre Fantasy

During the pre-processing, it was also noticed that the lemmatization was not very good as the past forms of the words were not taken back to its root version. For e.g. words like ‘drowned, returned, persuaded‘ were kept as it was, which could have been better, if it was lemmatized to ‘drown, return, and persuade‘ respectively.

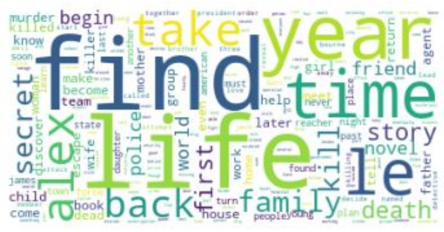


Figure 12: Word Cloud for genre Thriller

On the other side, the best accuracy on test data is around 69% with RoBERTa, while using TF-IDF, the maximum accuracy on test data was around 77%. In this case, a basic document representation technique is generating significantly better accuracy on test data than a complex transformer architecture. The possible reason for TF-IDF to outperform RoBERTa can be the small training data size. In this project, the RoBERTa model was not hyper tuned as well, which may have resulted in the low performance compared to TF-IDF.

It is also our point of interest that though both MLP and SGD give us the best accuracy, but time taken by SGD is extremely great which is less than 1 sec, while MLP takes around 310 sec.

Conclusion

Several different classifiers were used with the generated document representations using RoBERTa and TFIDF. With RoBERTa, it was noticed that the accuracy with MLP classifier and SGD classifier with TF-IDF were the best, however the time taken by SGD Classifier was much better. To conclude, the classifiers with TFIDF document representation along with MLP classifier is generating better accuracy on test data when compared to the RoBERTa model.

The data initially was highly imbalanced. In order to balance and predict well, the data was reduced a little. RoBERTa which is a deep neural network model requires a large amount of training data to learn meaningful representations. While using RoBERTa, bigger dataset may produce better results. Hypertuning the RoBERTa model may futher enhance the classification.

The results from TF-IDF model can be enhanced by using N-grams instead of individual words. The TF-IDF matrix in general is too large and sparse, thus dimensionality reduction techniques like Principle Component Analysis (PCA) or Linear Discriminant Analysis (LDA) can also help in improving computation time, memory usage and performance by removing noise and redundancy.

The conclusion of the comparison is sometimes simpler techniques give better results as we saw TFIDF gave us better accuracy than RoBERTa. However, It would be very interesting to see how well will these documentation representation techniques and the classifiers work with the suggested improvements.

References

Y. Bengio, P. Simard, and P. Frasconi. 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166.

Leo Breiman. 2001. Random forests. *Machine Learning*, 45(1):5–32.

Tianqi Chen and Carlos Guestrin. 2016. XGBoost. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Yoav Freund and Robert E Schapire. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139.

Andy Liaw and Matthew Wiener. 2001. Classification and regression by randomforest. *Forest*, 23.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. Cite arxiv:1907.11692.

Herbert Robbins and Sutton Monro. 1951. A Stochastic Approximation Method. *The Annals of Mathematical Statistics*, 22(3):400 – 407.

Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21.

Vladimir N. Vapnik. 1995. *The nature of statistical learning theory*. Springer-Verlag New York, Inc.