Improving classification through co-training approaches

Ragini Kihlman

A thesis submitted for the degree of **Doctor of Philosophy**

School of Computer Science and Electronic Engineering (CSEE)

University of Essex

August 2022

Abstract

Machine learning methods have found applications in fields as diverse as finance, health care, aviation, and social networking. Such methods rely on the availability of data and in particular labelled datasets to train algorithms to provide predictions. However, labelled datasets are difficult to source as annotating data is an expensive and time-consuming process. This thesis is focusing on improving the prediction for classification tasks through building on co-training which is a semi-supervised learning approach that trains two classifiers based on two different views of the data. We present an incremental approach to extending the co-training algorithm and deal with the lack of sufficiently labelled data. We develop a multi-label classification model that would classify the data collected by a domain-specific term extraction method based on Blums co-training binary classification model. To train the classifier in the domain of human rights, the method uses a labeled set compiled by experts as background knowledge. In the next step, we propose a method for randomly dividing the available features for applying matrix factorization so as to discover latent features underlying interactions between different kinds of entities present in a single view dataset. Using matrix factorization and similarity measures, the next method co-trains a large corpus of unstructured data to correctly classify it. This method evaluates each label and recommends documents with similar labels. In our final step, we implement two neural networks using two-view semi-supervised learning for text classification. This concept is extended to deep learning by using deep neural networks to train on different views of generated samples. This will calculate similarity in the probability distribution of predicted outcomes. Furthermore, the method adds noise to prevent it from affecting the classifier during prediction. The accuracy of classification is therefore improved by co-trained networks.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Normally, classifier systems are described in the context of supervised classification, and few studies have been carried out in the context of semi-supervised classification tasks. Although, there are important applications where semi-supervised multiple classifier systems are required, that are capable of analyzing both labeled data and unlabeled data, for instance multi-sensor remote sensing and multi-modal biometrics. Many researchers have been developing semi-supervised learning algorithms in the past decade, including S3VMs Ahmed et al. (2019), graph-based learning algorithms Chong et al. (2020), and disagreement-based learning algorithms He et al. (2020). Semi-supervised learning is a type of unsupervised learning that can benefit from utilizing unlabeled data without any human intervention. To classify web pages, Avrim Blum and Tom Mitchel introduced Co-training in 1988 in Blum and Mitchell (1998). A learning algorithm is based on the assumption that each of two sets of feature groups can be a representation of one dataset. Learning can be accomplished based on a representation as long as there is enough labeled data. For each view, two learning algorithms are trained separately, and new unlabeled data is added to the labeled data corpus based on the most confident of these algorithms' predictions.

Learning from labels and unlabeled data, co-training implicitly uses feature separation. As the feature sets are combined, incremental classifiers are built in incremental steps. Each classifier is initialized based on just a few labeled documents available. To build the labeled

set of samples, each classifier adds one unlabeled document from their class during every co-training iteration. As determined by the classification confidence from the underlying algorithm, documents with the highest classification confidence are selected. Following that, each classifier recreates its classifications based on the augmented labeled set. The posterior probabilities are multiplied together and re-normalized so that the prediction of the underlying classifiers is equal to one at classification time.

Training two classifiers using separate views of the same data provides the redundancy of the learning task in a weakly supervised learning paradigm called co-training. In previous studies regarding co-training, most of the studies assumed greater accuracy can be achieved by combining both views. Using this method, bootstrapping from a small set of labeled training data can be achieved through the use of a larger set of unlabeled training data. It is shown by, Pierce and Cardie (2001), that co-training reduces the error difference between co-trained and fully supervised classifiers by 36% in the case of labeled data as compared to unlabeled data.

Additionally, they stated that each view of the data should be sufficient to learn the classification task on its own and that the views should be conditionally independent. Using these assumptions, Blum also shows that any initial weak predictor can be boosted to arbitrarily high accuracy using unlabeled samples only by co-training, given a conditional independence assumption on the distribution D. Blum's conviction for co-training, however, could be compromised by three key questions.

- Could co-training be applied to learning problems without separating them into distinct views? The reason why co-training is so effective is due to the ability to split the data into two separate views. This method is useful when two distinct sets of features can be easily identified in the data. How well would the algorithm perform if the data or its features does not have a natural split?

- Furthermore, how does co-training scale when an extensive set of training sample is required in order to achieve useful performance levels? It is plausible to assume that,

because of mistakes made by the classifiers to add in incorrect labels as they iterated, the co-training algorithm will not scale well. In short, the classifiers may occasionally include incorrectly labeled instances in the labeled data. Co-training may become a problem if many iterations are required to learn the task, which will then impact the quality of subsequent view classifiers. Thus allowing for the possibility that co-training becomes less effective with time for large-scale learning tasks.

- Can textual data be assigned more than one label? Data may be extracted to analyze further, determine the context, determine treatment and rehabilitation, or extract information. Co-training frameworks are not capable of performing multi-label classification as in this approach each label represents a different classification with all the tasks related since they are meant for the same view. Blum's algorithm was not designed to handle multi-label classifications.

1.1 Maximize the potential of co-training

The co-training method on its own may not scale up or be suitable for situations with large amounts of unlabeled data and large-scale learning tasks, as well as data that cannot be divided into two distinct partitions. It is possible to extend and leverage co-training by adding extra steps at different phases of the training cycle, which then will be able to overcome many of the problems noted.

As a result, the motivation behind this thesis is to develop a model that would enhance and maximize the ability of the co-training algorithm to classify text effectively, regardless of whether it can be categorized into multiple labels. By introducing the following techniques, this thesis seeks to exploit the same principle to produce accurate predictions efficiently:

- **Multi-label classification -** This involves assigning a set of appropriate labels to training samples. In addition, the task is to learn a mapping from the feature space to the

actual labels in the label set. In general, getting labels for training samples is challenging and time-consuming, especially for multi-label learning tasks, where a variety of class labels need to be assigned to an instance. In spite of the fact that multi-label learning technologies have been widely used for many years, most of them require sufficient labels for training samples because the set of possible labels is so large. This idea was published successfully, Gokhale and Fasli (2017a), at the International Conference on the Frontiers and Advances in Data Science (FADS).

The co-training algorithm, being naturally capable of inductive classification, can be used to have each classifier independently trained on its own views of the data, thus augmenting the labeled training sets with its predictions that have the highest confidence. Following this, each classifier can be then retrained independently with the augmented training data. By exploiting the correlation between labels of labeled cases as well as maximum margin regularization over unlabeled cases, co-training can potentially be capable of assigning one or more labels to the sample. This process can be continued until the unlabeled samples are labeled.

- **Matrix factorization -** By using matrix decomposition, matrices can be reduced to their constituent parts, making it easier to calculate complex matrix operations. Methods of matrix decomposition are also known as matrix factorization methods as aptly demonstrated by Chen et al. (2016). Matrix factorization is commonly used when dealing with recommendation systems. The scenario corresponds to a group of m entities mapping to a group of n other entities. The movie recommendation for users is a very common example of a successful matrix factorization. By identifying the proximity and distance between a movie and a user, the representation of movies and users can be compared and matched. However, both representations need to be cast as vectors of the same dimension. This provides a matrix with information about the user, the movie, and the degree to which the user and movie interact. Thus, building an information representation of likely user movie combinations.

Combining this technique with Co-training algorithms is advantageous at the point where classifiers have to be trained on a set of the labeled and unlabelled samples. The single-view data set is randomly divided into two views based on the labels, with each view being capable of standing alone and successfully classifying targets. As part of the labeled set, the human expert will set a tentative rank value for each item. The factorization of labeled and unlabeled data is computed using the ranks along with Singular Value Decomposition. This enables the variable-level co-training model to include multi-label classifications constructed from label ranking patterns. A published version of this concept was presented at the 2018 IEEE International Conference on Big Data Gokhale and Fasli (2018)

- **Similarity recommendation -** Collaborative filtering or matrix factorization technique is widely used to recommend additional combinations of user and item in order to establish associations between the two. It is usually difficult for users to determine which items they want based on their interests and preferences. Therefore, recommendations are now based on the similarity between either the content or the users accessing it. By using a similarity matrix, they provide their customers with the next most similar product. This is to help the users search through a large space of possible items in a personalized way for items similar to those they have purchased previously.

  This method can be deployed on large unlabeled datasets to improve the quality of recommendations by harmonizing similarity measures with the co-training process. It is typical for any recommender systems to make recommendations based on ranking or some feedback given by users, but there are some situations when this feedback is inadequate. If sparsely labeled feedback data is present, matrix factorization can be used together with large unlabeled datasets to rate the data, forming an adequate set for the recommender algorithm. Once the new ranks have been identified, they are added to the labeled data thus improving the generated predictions. The 2019 IEEE International Conference on Big Data published a paper about this theory, Kihlman

and Fasli (2019)

- **Deep co-training-** Unlike classical machine learning, deep learning uses different kinds of data and learns using different methods. A machine learning algorithm uses structured data with labels to make predictions. This means that specific features are defined and organized as tables based on input data. Unstructured data is not necessarily excluded from this; it simply means it usually goes through some pre-processing to convert it to a structured format. A significant part of the pre-processing that is usually needed for machine learning is eliminated with deep learning. Algorithms like these can process unstructured data, for instance text or images, and automate feature extraction, removing the need for human experts. In this context, the co-training framework can be extended to include discovering and propagating labels to unsupervised samples, in order to improve classification performance using learned deep neural network weights rather than starting with random initial weights, and then further training the model on available labeled data.

  By using two neural networks and a deep-learning model to implement a co-training framework based on deep learning, we implement semi-supervised learning for text classification. Using two separate classifiers, the text is categorized according to two independent learner views, as with the original co-training. In order to extend this idea to deep learning, deep co-training models are trained on multiple views of generated samples in order to estimate the probability distribution of predicted outcomes. This co-training framework depends on co-trained networks in order to classify text into multiple labels. Co-trained networks, therefore, provide more relevant information about the data. In 2021, this was presented at the IEEE International Conference on Big Data, Kihlman and Fasli (2021)

1.2 Scope of the thesis

The ability to make accurate predictions is crucial in data streams contexts that require them for planning and decision-making. Ariyo et al. (2014) showed that researchers should build better predictive models. There are situations where large quantities of data cannot be labeled because label assignment is impossible, it is expensive, or the job is time-consuming. Unlabeled data is commonly found in Engineering Systems (video object detection) like Rosenberg et al. (2005), Physics (weather forecasting and ecological models) in Imai et al. (2015), Biology (cellular models) discussed in Uslan and Seker (2016), and Economy/Finance (stock price forecasting) in Ariyo et al. (2014). A majority of these areas rely on real-time processing of data from streams show in Zhou and Li (2007).

With binary classification and document classification allowing two distinct views on the documents, co-training has proven popular. Co-training, however, can generate predictions using more than one model at the same time, thus extending its effectiveness. By varying the inputs and methods of regression or by changing parameterization, as explained in Zhou and Li (2007) the model diversity can be created.

By gradually introducing the following concepts in this thesis, it is recognized that the co-training framework is extensible and is capable of enhancing prediction accuracy. The goals that this thesis intended to address include:

- The ability to create a labeled training set with minimal input from experts using an unlabeled corpus of unstructured and unlabeled text for a multi-label classification model using co-training.

- By adding additional classification features to this multi-label model, it becomes possible to classify data effectively even if two distinct sets of data cannot be derived. This will avoid redundancy in label predictions between the two classification methods

- Detection of hidden patterns between data and labels to construct a recommender

system to be able to select the best predictions based on their confidence levels.

- Develop neural networks that can cope with even larger and more complex data-sets by exploring deep learning models

The research domain and the focus of this thesis is human rights violations, where the prototype system is able to correctly classify victim survivors' stories based on an initial set of labelled violations annotated by human rights experts.

1.2.1 Human rights need technological advances

Over the past few years, human rights violations involving torture have been on the rise. Charitable organizations offer corrective therapy, support, and the care needed to help victims of abuse and torture rebuild their lives. The researchers collect data about victims and their stories so that reports can be compiled and published primarily in areas of political science, sociology, international law, and human rights, to influence governments, institutions, and others to make positive decisions and promote human rights and justice around the globe as shown in HRDAG (n.d.); Project (n.d.); WITNESS (n.d.). In spite of this, many of these organizations lack the capacity to fully utilize the data collected, recorded, and subsequently analyzed to improve their services to victims of torture as detailed in Meyer (n.d.). For human rights organizations and non-government organizations assisting torture victims for the recovery and rehabilitation process, they need powerful, sophisticated tools to recognize patterns and insights within these datasets. The method involves using human rights experts' background knowledge to extract the terms for labeling a small set of data. By using the labeled and unlabeled samples together, additional labels can be generated for the remaining dataset. Several tools using classifiers and heuristics have been proposed for automatic domain detection and text classification in articles Portugal et al. (2018). To provide the most appropriate type and level of support, it is essential to identify what form of torture or abuse a victim has been exposed to. Counselling, legal, and medical support

are provided as part of the rehabilitation and treatment of human rights abuse victims. In some cases, it still takes many passes, is highly manual, and is time-consuming to determine whether a person has been tortured. Unstructured and typically large data sets are received. Researchers have tried to structure the text by using text analysers and data mining tools. It is still, however, a primitive state in which it needs manual intervention for it to become usable as valid data in the human rights domain as demonstrated in HURIDOCS (n.d.). There has been a long-standing problem of human rights organizations collecting data and typically being unable to analyze it, and as a consequence not being able to use that data to its full potential as explained in Rodrigues (2020); Thinyane and Sassetti (2020). Creating a well-structured database for analyzing human rights survivor stories is still a very time-consuming and complex process and it can be emotionally challenging for the people involved. New technologies have made it possible to access high-quality data to aid in further research and study. By using two neural network classifiers, the extended co-training algorithm addresses the challenges in the human rights domain by classifying survivor stories into one or more appropriate categories (abuses) to enable more effective and targeted support rehabilitation to be offered and care for the victims.

1.2.2 Research intentions

In the section above, the emphasis of this thesis was laid out. In machine learning and data mining, the expectation is to achieve better prediction results, while being computationally efficient as well. Following are the research questions posed so that they can assist in providing evidence that the solutions provided align with the goals and objectives of this thesis. They are:

- Is it possible to categorize large unstructured data sets with minimal manual labeling from experts? Could human rights survivors' stories be categorized into multiple labels based on the violations they suffered, allowing the caseworkers to identify the rehabilitation and treatment schedule for each survivor before they are told about it?

- How likely is it that the learning model will be able to classify a new story into one or more categories? Based on features alone, some survivor stories cannot be divided into two groups. How well would the learning model perform if it were able to look beyond the overlap between the two classifiers and predict correctly?Introduction So as to minimize the chances of adding incorrect labels and deteriorating the labeled set for further iterations.

- With the help of co-training, can we design a continuous, efficient, and cost-efficient system that recommends documents with similar labels? Caseworkers will do this when seeking out existing case studies with similar labels in order to find out what rehabilitation was provided and match it with the case of a survivor who experienced similar circumstances. Consequently, the survivor receives assistance in a timely manner.

- Would neural networks be able to implement the same co-training model on complex datasets? Can co-training two neural networks identify hidden patterns in the data that are not influenced by noise and bias in cases where there is a chance that the survivor stories might be skewed by noise and bias? What is the likelihood of neural networks predicting correct labels when the data has been augmented with noise?

In my view, the validation of the above claims has several important positive consequences, which align with my motivation for pursuing my thesis on human rights violations. Such implications would have a significant positive impact on research in the human rights domain, as well as any field with a large and disorganized dataset.

- Question 1 suggests that the reduction in the contribution of experts (in this situation, human rights experts) will result in a substantial reduction in the cost of manual annotation, thus facilitating a prompt turnaround of results. It will be less expensive to obtain human rights experts' contributions for one labeled set and then expand that set further using co-training than to have them completely label a large corpus. A

method that combines a labeled set with additional labels from an unlabeled set will be quicker than a method that creates clusters based on unsupervised learning.

- As a result of question 2, researchers can assign documents to more than one category. The specific case of human rights would benefit by understanding the victims holistically and administering appropriate treatments and rehabilitation

- Question 3 suggests moving away from standard supervised and unsupervised approaches and venturing into semi-supervised learning frameworks that would classify new documents and update the already labeled set with new labels that are not already there. For precise classification, this methodology would employ more than one classifier at the same time. With this approach, a small, labeled sample can be used to create a large corpus of labels, which can then be used to efficiently classify unstructured data.

- Lastly, question 4 proposes that two neural networks be trained together to reveal hidden patterns and relationships using only a small labeled data set as input.

1.3 Experimental work

The approach taken in this thesis is a step-by-step process of extending the co-training algorithm, including the above-listed techniques at intervals to judge how well it performs. The goal of this thesis is to identify human rights violations from survivor stories in order to provide survivors with the rehabilitation and services they need for their return to society. Experimental data on survivor stories is collected to figure out the performance of the co-training algorithm. The data will be used throughout all phases of its evolution along with the same metrics used to gauge its efficiency and accuracy. But this evolving algorithm is not meant to be domain isolated, but to be applicable to any domain providing a minimally labeled dataset for it to learn from. For this reason, real-world datasets are used for gener-

ating results and checking how well it handles these scenarios. In the following section, we will describe the datasets used to evaluate the algorithm. To evaluate the models described in the above section, this thesis will use two types of datasets. One is an experimental dataset collected by scraping the internet and the other is a real-world dataset for an e-commerce application. The two datasets will be used to test how the models perform in various scenarios and identify any flaws or drawbacks that can be fixed to make the framework more helpful in making predictions.

1.3.1 Experimental dataset

The experimental datasets are two datasets of human rights violations A downloadable 10 MB set of unlabeled testimonies provides 400 survivor stories culled from the Internet and stories_of_victims_of_torture data set from Business and Centre (n.d.). In this dataset, 9000 stories about Xinjiang victims are collected. This dataset is publicly available for download on Kaggle and on their website.

The data has been pre-processed to make it suitable for classification. As part of the normalization process, stop words and punctuation are removed from the text, and all numbers are rounded to the nearest whole number. Additionally, the top 50 features were generated with the TfIdf vectorizer. Labels have been manually annotated on this data, and each story is further classified according to one or more of these labels.

1.3.2 E-commerce dataset

Bonusway, is a Finnish online service that pays a purchase credit for e-commerce purchases made by its users. to extend their vision of partnership marketing, Bonusway intends to gather knowledge about its users and then be able to rank the increasing amount of offers for each user, to optimize what offers they show each user in their online feed. They implemented an in-house recommender system to improve sales. However, they face some roadblocks with this approach. The challenges faced by them were:

- data scarcity issues where there will be no data directly connected to any offers for the majority of the users;
- A good number of the users have not made any purchases or clicked on any link, a typical cold start issue.
- There are always new offers on items, but no relationship can be established between user purchases and offers.

The objective of the algorithm proposed in this thesis was to overcome most of the challenges faced by Bonusway and hence was deployed in their production environment to evaluate the effectiveness of the personalised recommendations.

The dataset used for the recommender system is a collection of the following based on each country -

- all active stores that generate offers
- user-specific click events for e.g., click on links in the email, store links, offer links
- purchases made for an offer on a store
- user data, specifically gender, birth year, and country of residence

The tests were run on a total of 4.2 million unique users, spread over 11 countries. The smallest country had 13 thousand users and the largest 2.4 million users. The test to control ratio was 9:1. The tests were split over 14 email events spaced roughly one week apart. The test and control groups were also chosen to have an equal ratio of gender, age, and activity (clicks, purchases). The control group were sent randomly chosen offers.

1.4 Structure of the thesis

The thesis attempts to address the four main thesis questions by exploring the existing co-training framework and extending and enhancing it in order to answer the earlier questions.

As the answers to the questions are incrementally added, the algorithm reaches its final state and remains true to the principle Blum demonstrated in his co-training model. This combination of classifiers allows us to detect hidden patterns within articles as well as find similar articles. This makes it easier to generate accurate predictions. In structuring the thesis, a similar approach has been taken. Chapter 2 discusses the current research and background related to this thesis. Several machine learning methods are compared, including supervised, unsupervised, and semi-supervised. A key component of Chapter 3 is transitioning Blum's binary classification method to use the confidence in prediction probabilities for multi-label text classification, which provides a first step in developing the co-training algorithm. Chapters 4 and 5 discuss Latent Features and Ranks that are used to classify text by matrix factorization and similarity measures. It is demonstrated in Chapter 6 how to transfer a co-training method to a deep learning model, in which neural networks replace classifiers in the training process. The methods use the same dataset explained in this chapter, but each of them has its own evaluations and results in order to test the method's performance, as well as comparing them with state-of-the-art methods for text classification and the previous versions of the extended algorithm. Lastly, chapter 7 compares and contrasts all methods described throughout the thesis.

Chapter 2

Machine Learning with text classification techniques: a review and analysis

As a result of the digital revolution, many organizations have begun capturing large volumes of historical data about their operations, products, and customers. Researchers have been collecting more complex experimental datasets in many fields, including gigabytes of MRI (magnetic resonance imaging) data, where the authors,Khalid et al. (2020), have reviewed various machine learning methodologies that improve disease detection in brain cells, movie ratings by viewers summarized in Khan et al. (2020), improving customer experience using online shopping, and social media comments as seen in Stoyanova et al. (2020). Using data mining, we are able to uncover general patterns in the historical data and improve the process of making decisions. Additionally, factors like the falling cost of large data storage devices and the growing ease of collecting data over networks, machine learning algorithms that are robust and efficient, and a reduction in the cost of computing power have all contributed to the use of computationally intensive methods for analyzing data.

Singh et al. (2020) points out in his article that a variety of factors may cause a dataset to be messy or noisy, including missing observations, confounding factors, and outliers. It is probably necessary to perform post-processing on the data before fitting the model. Thereby

gain insight from data and transforming data into knowledge. This is achieved by deploying data mining and machine learning methods, which is therefore one of the objectives of this thesis.

As part of data science, machine learning plays a crucial role. With the help of statistical and computational techniques, data mining projects can uncover valuable insights using algorithms that predict or classify. In computational science, ML (machine learning) is viewed as a subset of artificial intelligence (AI) as it is a method of understanding and analyzing data patterns and structures to enable outside-the-box decision-making, learning, and reasoning. As explained in Matloff (2017) machine learning use cases can be divided into these broad categories:

- Classification - where large datasets can be broken down into meaningful subsets using classification models. Image recognition in Fasanmade et al. (2020) and natural language processing in Müller et al. (2020) are some examples.

- Regression - Predictions can be made using regression models. It could be used, for example, to forecast sales based on macroeconomic in Adams et al. (2021) and weather forecasts in Hewage et al. (2021).

- Clustering - Collectively arranging samples of homogeneous meaning and connotation into a single cluster. Organizing documents according to their content in Sarddar et al. (2020), for example.

- Association rule learning- Discover potential patterns of association in data, for instance, if you buy hamburger buns, you're likely to buy hamburgers and/or cola as well as explained in Akas et al. (2020)

- Structured output - Use related items as a basis for constructing a complex output. For example, natural language parsing trees, or image recognition bounding boxes as shown by Liu et al. (2021).

- Ranking - Determine the position on a scale by evaluating ratings and feedback. An example would be a movie review based on user feedback as shown by Jani et al. (2020)

As data collection continues to grow,an increased computational power would be needed for better processing and sorting this data. Therefore each of these components can result in research topics to address questions. While there are many applications that use machine learning like Image recognition, Speech recognition in Park et al. (2020), traffic prediction in Boukerche and Wang (2020), Virtual personal assistant demonstrated by Arora et al. (2021), the area of focus for this thesis is mainly on classifying text. The process of classifying texts consists of extracting a set of generic tags from the unstructured text. These generic tags come from a set of pre-defined categories.

As a result of classifying large textual data, it is possible to standardize the platform, make searching easier, and simplify navigation so that users have a better experience. It is also an effective tool for extracting value from unstructured data. Text classification of labeled documents has seen a growth with the advent of more complex data collection and readily available data sets. Today many domains like, for instance, medicine by Motwani et al. (2017), finance by Patel et al. (2015), and traffic regulations by Nallaperuma et al. (2019) make use of data mining, machine learning and automation techniques for pattern matching, cognitive analysis, and reasoning.

The success of any machine-learning algorithm is driven by three basic components, they are -

- Making decisions based on a computational algorithm.
- Variables and features that influence the decision.
- Knowledge for which an answer already exists that facilitates (trains) the system to learn.

Machine learning algorithms for text classification can be divided into three categories based on these components; Supervised, Unsupervised ad Reinforcement learning. In the following paragraphs, we explain how machine learning can be divided into various categories along with the need of having a hybrid system that would efficiently obtain desired results.

2.1 Supervised Learning

Text classification plays an important role in information extraction and summarizing, text retrieval, and question answering. The objective of a classification task is to identify whether input data belonged to a group, class, or category of data. In some classification techniques, classifications are based on significant words or features extracted from text documents and these extracted features are then assigned classes. This kind of learning is termed as supervised learning as the learning happens from a training set that has classes that are predefined, either by an expert or a knowledge base. By supervised learning, we mean that the algorithm is learning from the training dataset using these predefined classes. This thesis focuses entirely only on the classification method, discussed in the section above, where the algorithm tries to label input into one or multiple distinct classes. In other words, we are not attempting to predict a continuous outcome, as seen in regression analysis, but rather, attempting to predict one or more discrete values in order to classify a large and unstructured dataset. Using classification you can approximate the mapping function (f) from discrete input variables (X) to a discrete output variable (Y). In the paper Saravanan and Sujatha (2018), the authors have reviewed and identified algorithms that efficiently classify documents using supervised learning. those that are particularly relevant for this thesis and those that will be compared to the proposed algorithm are:

- Naïve Bayesian model - Under the simple assumption that the attributes are conditionally independent, Naive Bayes is a probabilistic classifier inspired by the Bayes

theorem. All the predictors are assumed to have equal influence on the outcome under this method. It assumes independence among child nodes separated from their parent to predict the probability of an object as described by Huang and Li (2011). Naive Bayes takes linear time rather than expensive iterative approximations to obtain good results on classification tasks. However, Naive Bayes models are constrained by a problem known as the zero probability problem. If the conditional probability for a particular attribute is zero, the prediction is invalid and further estimation techniques must be applied to give valid predictions.

- Decisions trees - Kotsiantis (2013) states that using this method, instances are sorted based on the feature value and predictions are made using simple decision rules established from training data. In a decision tree, models are built in a tree structure and classification is based on a set of if-then rules that are mutually exclusive and exhaustive. Each rule is learned sequentially using the training data. The tuples covered by the learned rules are removed each time the rule is learned. This process is continued on the training set until meeting a termination condition. A decision tree can easily over-fit itself, generating too many branches and reflecting outliers or noise anomalies causing the model to have a very poor performance. In order to maintain efficiency and avoid overfitting, additional tasks for instance pruning and post-pruning have to be added. Pre-pruning stops tree growth early, and post-pruning removes the branches from a fully grown tree.

- Support vector machine (SVM) - As discussed by Dalal and Zaveri (2011), this algorithm ensures that we can sort n-dimensional space into classes by creating the best line or decision boundary. This allows us to easily place a new data point into the correct category in the future. This decision boundary is called a hyperplane. SVM is one of the popular supervised learning algorithms. Being memory efficient, SVM works well with high dimensional spaces but it has difficulty coping with large datasets. There is often a problem of overfitting because these datasets contain noise

and overlapped classes. In order to select the optimal kernel for separating the data points on a hyperplane, several memory-intensive methods are required.

Although supervised learning has proven to be effective and efficient, the process tends to generate inaccurate results. According to what was said by Saravanan and Sujatha (2018), some of the major flaws seen using this approach are

- The goal of supervised learning is to build a knowledge base to efficiently classify a dataset using the prior information. If no such information is available, supervised learning lacks the ability to discover the attributes of data on its own or to leverage diverse information in the data. This means it cannot handle some of the more complex tasks involving large datasets. This results in longer training computation times.

- A good supervised classifier depends on good labeling and data annotation. Manual labeling is very costly and tedious because it is usually done by experts. It may be difficult to obtain data in highly specialized fields for large-scale labeling. Moreover, even if the data is obtained, it may be false or incomplete for any number of reasons. The process of data annotation on its own is time- and cost-consuming

- The need to produce insights from large datasets has attracted both academia and industry to data analytics. Interestingly, Tsai et al. (2015) thinks that uncertainty exists in every phase of data analytics and comes from many different sources, including data collection (e.g., environmental fluctuations and sample bias), concept variance (e.g., analytical objectives differ from one another) and multimodality (e.g., complexity and noise generated by such sensors as numerical, text, and image data).The computational efficiency or scalability of supervised learning is insufficient to handle both the complexity and uncertainty of data (e.g., large volumes, high speeds, varying types, low value density).

- The collection of large amounts of data can lead to data distortion due to noise and signal interference. As a result of this, there are a lot of outliers that cause the su-

pervised learning methods to learn on the noise rather than the actual data features. Resulting in overfitting that negatively impacts the performance of the model on new data.

2.2 Unsupervised Learning

While supervised learning requires neatly labelled input set for the training data to learn from, Dey (2016) explains that Unsupervised learning draws inferences from datasets consisting of input data without labelled responses. By analyzing similarities in a dataset, unsupervised learning seeks to discover the underlying structure, hidden patterns, and insights. There are no labels given to the learning algorithm, and it is left to find structure in its input on its own. The aim of unsupervised learning can be to find hidden patterns in data (discovery) or to gain knowledge (feature extraction). Unsupervised learning problems can be categorized as either clustering or association problems. Some of the common learning approaches of these methodologies are:

- Clustering - According to Liu et al. (2003), clustering involves comparing unlabeled data and grouping them together based on similarities or differences. Clustering algorithms take raw, unclassified data objects into groups defined by structure or pattern. Clustering algorithms can be divided into four types: exclusive, overlapping, hierarchical, and probabilistic. Cluster granularity can be fine-tuned by adjusting the number of clusters the data. Prominent clustering examples include Kmeans clustering by Wu (2012), Principle Component Analysis as shown by Jolliffe and Cadima (2016), and the latest Singular Value decomposition described in Murty et al. (2011). High-dimensional data features have a poor clustering effect and a high degree of complexity. During random sampling of data, many parameters are susceptible to spatial differences, resulting in inaccurate sampling to generate the clusters. Due to the increasing time required to compute these clusters, the algorithm performs poorly.

- Association rules - The association rules check for dependencies between data items and map them accordingly. Using the dataset, it searches for interesting relationships or associations among the variables. Using different rules, it identifies the interesting relationships between variables in a database. Support and confidence are used to measure the quality of a given rule, in terms of its usefulness (strength) and certainty. For example as explained by Ahmad and Doja (2013) using Apriori algorithm and FP growth. The appropriate parameters must be specified in advance to discover associations between them to generate rules. This is slow, inefficient, and uses a great deal of resources because it scans the database many times, generates a lot of candidate sets, and checks each of them.

Amongst challenges that unsupervised learning faces, as reviewed by Saravanan and Sujatha (2018), the ones this thesis would like to focus on are:

- A lack of prior knowledge makes it impossible to obtain precise information about data sorting in unsupervised learning. The machine requires it to do this on its own. Complicated computational tasks are created as a result of a large volume of data that must be sorted and categorized. Leading to longer training times.

- However, humans are still needed to validate the output parameters even if the algorithms find patterns in the data without their involvement. For example, an unsupervised learning model might reveal that online shoppers often buy groups of products at the same time. However, the data analyst will need to assess whether it is appropriate for the advisory service.

- Clustering data when clusters are of varying sizes and densities can be challenging. This is due to the fact that spectral classes do not always correspond to informational classes, as well as the lack of transparency for the basis for clustering data.

2.3 Semi-supervised Learning

Semi-supervised learning generally involves both labeled and unlabeled data in a training dataset. The method is especially useful for extracting relevant features from data when extracting them can be difficult, and labeling samples can be expensive as well as time-consuming for experts. The manual labeling of all the data would take too long and cost too much - but a model that learned from a small amount of labeled data could improve its accuracy compared to an unsupervised model. This is semi-supervised learning. The algorithm solves classification problems, which means you need a supervised learning algorithm to complete the task. At the same time, you want to train your model without using labels for every single sample, a task that's handled by unsupervised machine learning.

Acknowledging the growing concern of having insufficient labeled dataset to label and organize extensive unstructured data collected over the Internet in several domains, Blum and Mitchell (1998) attempt to increase the amount of the labeled dataset using large amounts of the available unlabeled data with their proposition of the co-training algorithm. The co-training algorithm works by generating two classifiers trained on the input labeled data, which are then used to tag new unlabeled data. From this newly labeled data, the most confident predictions are added to the set of labeled data. Awasthi et al. (2021) applied Co-training to combine labelled and unlabeled data and was successfully able to decrease labelling costs. Ma et al. (2017) have demonstrated an improved co-training algorithm which trains the classifiers under two views of data in a serial way and alternatively updates the pseudo labels of unlabeled data to improve the performance of classifiers in each training round. It has generally been found to bring improvement in cases where no additional unlabeled data are used.

The efficiency of the co-training algorithm per Blum and Mitchell (1998) relied on the conditional independence of the views that the training data could be split into. However after Blum, Abney (2002) demonstrated that overlooking the conditional independence of the views still maintained the effectiveness of the co-training algorithm. This was achieved

by deploying a greedy algorithm that works under a weaker independence assumption to produce good results in labeling the data. This works as an added advantage for the model this paper is trying to put forth as it is extremely challenging to split a survivor stories into two conditionally independent views. In the sections that follow, this thesis would like to show how to achieve new labels to tag unlabeled stories by relaxing the criteria and dividing the training dataset based on the number of discovered features.

2.4 Deep learning

The field of Deep Learning focuses on algorithms that are inspired by the structure and function of brains, known as artificial neural networks. Andrew Ng describes the idea of deep learning the following way in his 2013 talk Huang (2013):

"Using brain simulations, hope to: Make learning algorithms much better and easier to use. Make revolutionary advances in machine learning and AI. I believe this is our best shot at progress towards real AI" Computer systems with nodes that operate like neurons do in the brain are called neural networks. A study , Kosko (1988) published in 1998 demonstrated that By recognizing hidden patterns and correlations in raw data, they can cluster and classify it, and - over time - continuously learn and improve. Originally, neural networks were designed to simulate the functionality of the human brain. Researchers diverged from a strictly biological approach after they began to use neural networks for specific tasks. Yao (1993) researches on potential interactions between connectionist learning systems, i.e., artificial neural networks (ANNs), and evolutionary search procedures. Neural networks are now used for a multitude of tasks, including video and board games Gambus and Shafer (2018), speech recognition Bell et al. (2020), machine translation Stahlberg (2020), and social network filtering Rivas et al. (2020).

There are different kinds of deep neural networks, and each has limitations and advantages depending on its purpose. Among the examples are:

- Feed forward neural networks, each perceptron in one layer is connected to every perceptron in the next layer. A layer is only able to feed information forward in one direction only as seen in Haldorai and Ramu (2021)

- Convolutional neural networks (CNNs)- Five types of layers are present in convolutional neural networks. The layers have different functions, for instance summarizing, connecting, and activating. CNNs have also been applied to image processing Han et al. (2020), and forecasting Livieris et al. (2020).

- Recurrent neural network (RNN) - uses sequential information to perform its computations, for instance time-stamped data from a sensor device or a spoken sentence, composed of a series of terms. Forecasting and time series applications Hewamalage et al. (2021) use RNNs.

- Autoencoder neural networks are used to create abstractions called encoders from a set of inputs. Depending on whether the abstraction is linear or nonlinear, a classifier can then use it Huang et al. (2020)

This thesis investigates whether two neural networks can classify a minimally labeled training set simultaneously and learn over time by using CNNs with co-training.

Feature learning, also referred to as automatic feature extraction from raw data, is another benefit of deep learning models, in addition to scalability. Bengio (2012) outlines how the algorithms in deep learning can discover and learn good representations by learning features. On problems in which the inputs (and outputs) are analog, deep learning excels. This means they do not represent a few quantities in a tabular format, but rather are images of pixel data, documents of text data, or audio files.

Albawi et al. (2017) present a powerful artificial neural network technique known as convolutional networks. These networks preserve the spatial structure of the problem and were developed for recognition tasks for instance handwritten digit recognition. Due to their ability to solve complex computer vision and natural language processing problems, these

programs are becoming very popular. As with multilayer perceptron feedforward neural networks Cinar (2020), this technique is achieving great success because it scales as data and models grow and can be trained with backpropagation. The selection of text feature items is a fundamental aspect of text information mining and retrieval, as stated in Bengio (2012). The traditional method of extracting features requires handcrafted features. It is a lengthy process to design an effective feature by hand, but deep learning enables to learn new effective feature representations from training data as proposed by Wang and Raj (2017). Deep learning has made significant contributions to text mining as a method of identifying features. Deep learning automatically learns features from data, rather than adopting handcrafted features, which rely mainly on prior knowledge and are highly inefficient when dealing with data. A key aspect of deep learning is that these layers of features are not designed by humans, they're learned from data by applying a general learning procedure as mentioned in Singh et al. (2013). Since deep learning does not require much engineering to implement, it can easily take advantage of the increased amount of available computation and data. Deep learning can automatically learn feature representation from data, including millions of parameters. Deep learning has three types of learning: supervised, unsupervised, and semi-supervised.

Unsupervised and supervised deep learning approaches discussed in Liang et al. (2017); Kowsari et al. (2017); Singh et al. (2016); Chowdhury and Saha (2005) are becoming more popular as data and artificial intelligence gain traction. Moreover according to Nguyen et al. (2019), this resulted in the creation of exceptional deep learning algorithms that can use multiple processors and graphics processing units and handle very large datasets. Several studies have shown that neural networks outperform standard machine learning algorithms as early as 1996, when Nguyen et al. (2019) compared the efficiency and performance of neural networks versus machine algorithms. While medical data can be complex, it seemed that neural networks could provide better results as they were cost-effective. This high performance can only be achieved via manual labeling or annotation of the data. In general,

current machine learning algorithms are hard to train because there does not exist labeled, annotated data that can be used for in-depth research. Additionally, according to the survey presented by Pant et al. (2019) a number of supervised machine learning algorithms have difficulty efficiently classifying large databases into multiple labels. It is always a dilemma to choose between efficiency and accuracy. Given the massive amounts of unlabeled text-based data that have been collected over the years by various organizations, the use of text-based data sets would be an interesting option when building a labeled dataset.

One major challenge in Data Analytics is dealing with fast-moving data streams. An analysis of this type is helpful to monitor tasks, for instance fraud detection. Deep Learning should be adapted to handle streaming data in order to deal with large amounts of continuous data input. In their paper, Zhou et al., in Zhou et al. (2012), show how denoising autoencoders can be used for incremental feature learning on large datasets with a Deep Learning technique as described in Bengio et al. (2013). These denoising autoencoders operate on corrupted inputs to produce features. The extracted features are robust to noisy inputs and can be used for classification. Despite its potential, High Dimensional Data Analytics is largely unexplored. Marginally stacked denoising autoencoders (mSDAs), which are more computationally efficient than regular stacked denoising autoencoders (SDAs), have been introduced in Chen et al. (2012). However, to address the high-dimensionality found in some Data domains, it is still necessary to develop deep learning-based solutions, either by adapting or by developing new approaches. When analyzing data with Deep Learning algorithms, there is another important issue to consider: whether to utilize the entire Data input corpus. It is generally thought that Deep Learning algorithms are applied to train high-level data representation patterns from portions of the input corpus, and then use the remaining input corpus with the newly learned patterns for extracting the abstractions and representations of data. Hence, it is necessary to conduct research in order to identify what are the optimal input corpora from all Data to develop a computationally efficient deep learning algorithm.

The input corpus of some Data applications can include both labeled and unlabeled data, for example, in cyber-security as discussed in Suthaharan (2014), fraud detection put forth by Wang et al. (2013), and computer vision in Freytag et al. (2013). Such approaches can rely on semi-supervised training methods towards defining criteria for good data representation learning using Deep Learning algorithms. Having learned representations and patterns from the unlabeled/unsupervised data, the available labeled/supervised data can be used to further tune and improve the learned representations and patterns for a specific analytics task, including semantic indexing or discriminative modeling. Data mining variants for instance active learning, which incorporate feedback from crowd-sourcing or human experts, can also help obtain improved data representations as the labels on some data samples can be used for improving learned data representations.

2.5 Multi-label classification

Depending on the complexity of the classification, three different types of classification can be conducted: binary classification, multi-class classification, and multi-label classification.

- Binary classification: This is used when there are only two distinct classes and the data being categorized belongs exclusively to one of those classes. For example, to identify if a post about a given product is positive or negative; as reviewed in Kumari and Srivastava (2017)

- Multi-class classification: As surveyed by Aly (2005) when there are three or more classes and the data we want to classify belongs exclusively to one of the classes, for example to determine if a semaphore on an image is red, yellow, or green;

- Multi-label classification: This type of classification, as told by Tsoumakas and Katakis (2007) is used when there are more than two classes and the data we want to classify may belong one or more classes simultaneously, for example to scene labeling on an image.

The purpose of this thesis is to make large unstructured datasets more easily classifiable by machine learning techniques. We performed experimental analysis on a dataset about stories of people who had suffered human rights violations as part of our study in order to determine whether the combination of multiple classification algorithms yielded better results. Therefore we shall focus mainly on the multi-label classification. Multi-label classification is a result of analyzing text categorization, in which documents may belong to more than one category at a time. A multi-label classification algorithm consists of a training set of instances that each have a set of labels, and the purpose is to predict the labels of unseen instances by analyzing the training set with known labels. In contrast to traditional classification that uses mutually exclusive labels, multi-label classification uses specialized machine learning algorithms that predict multiple mutually classes. As seen from recent studies, text classification is the primary scope for multi-label classification techniques as talked about by Gonçalves and Quaresma (2003); Joachims (1998); Luo and Zincir-Heywood (2005); McDonald et al. (2005). However, it is also finding significant applications in areas like bio-informatics in Clare and King (2001); Zhang and Zhou (2005); Elisseeff and Weston (2002),medical diagnosis in Karalic and Pirnat (1991), and scene classification in Boutell et al. (2003); Shen et al. (2003). As a result, Machine Learning techniques have the potential to improve methods that require a multi-label classification by deploying the exploitation of label dependencies. To improve the analysis of victim survivor stories using multi-label classification, this thesis will focus on using a multi-label classification approach.

2.6 Classification and recommender systems

Scientists are increasingly using classification and recommendation systems to diagnose and plan future treatment. We will discuss how classification and recommendation techniques can serve as a basis for improving machine learning.

Lee et al. (2005) provided a comprehensive discussion of how recommendation systems work, focusing on a collaborative filtering. Using this filtering, it shows different algorithms based on the profile of the user and on the characteristics of the product. Additionally, several classification methods are described that may be used as inputs for recommendation systems. In Adomavicius and Tuzhilin (2005) a general overview of recommendation systems was presented. In this study, they sought to gain a better understanding of the current development in recommendation methods, especially the use of collaborative filtering.

Wu and Ester (2015) illustrated the challenges of recommending based on user preferences. Using a combination of recommendation systems and classifiers, the authors identified words that indicate a gap between users' expectations and their actual experience. According to them, traditional recommendation systems analyze past classifications. Such systems consider the users' preferences history; recommendations made from opinion classifications, however, consider the existing evaluations in the moment. The use of machine learning techniques to make recommendations has grown in importance in the research community, with many papers on the topic. Some of the proposals make use of lexical classifiers to detect possible emotions using content-based recommendations, like Nilashi et al. (2016). There are many authors who have explored the traditional branch of ML, using methods for instance logistic regression, the Pearson correlation coefficient, or the naive Bayes theorem based on probability, among others. The authors of this paper focused on making extensions of these methods to solve problems inherent in recommendation systems for instance cold start or scalability. Since the beginning of recommendation systems, the cold start problem Schein et al. (2002) has been a concern since a system cannot guarantee accurate results in recommending if it does not have enough data. Problems like this tend to arise more often during the implementation of a new system when there is no data. A significant challenge has arisen in terms of scalability due to the significant growth in information over the years and the volume of data that must be managed. When very large data sets are involved, both product- and user-based recommendation systems affect performance and accuracy.

Another noteworthy work that uses ML to assess sentiment classification is Kharde et al. (2016). A comparative study of the three approaches presented in this work concludes that supervised ML presents a greater level of accuracy, whereas lexicon-based approaches are also competitive, since they require less effort and they are not affected by the size and quality of the training data sets. The study presented by Mu (2018) provides a comprehensive review of deep learning-based recommendation systems. This work concludes with a summary of future research lines for instance cross-domain, scalability, explainability, or deep composite model-based recommender systems.

We performed experimental analysis on a dataset about stories of people who had suffered human rights violations as part of our study in order to determine whether the combination of multiple classification algorithms yielded better results.Human rights violations in terms of torture have been an area of concern for many years now. Many charitable organisations reach out to victims of abuse and torture to provide corrective therapy, support and the care needed for their rehabilitation. While doing this, they collect data about the victims and their stories in order to collate and publish reports mostly in the fields of political science, sociology, international law and human rights so as to leverage government organisations, institutions and others to take corrective action, make changes and promote human rights and justice around the world HRDAG (n.d.); Project (n.d.); WITNESS (n.d.). However, due to the way in which the information is collected, recorded and subsequently analysed many of these organisations are yet to fully utilise the data collected to improve the services they offer to the victims Meyer (1996).

2.7 Evaluating the classification algorithms

The ability to decide whether or not our approach is working can be determined by evaluating the performance of the model. We can continue exploring and seeing how far our existing concept of the problem we are dealing with can go. In addition to revealing whether a

model is working, it can also reveal if our approach is not working, as a model that is failing at what it is supposed to do indicates that we misunderstood the data or the situation we are modeling.

It is crucial to select performance metrics appropriately during the evaluation of classifiers.There have been several performance metrics proposed for various applications. Typically, accuracy is measured by the percentage of correctly classified test instances. The accuracy of classifiers is evaluated by its precision and recall as shown by Ben-David (2007) and is widely applied in information retrieval as put forth by Baeza-Yates et al. (1999); medical decision makers prefer area under the receiver operating characteristic (ROC) curves (i.e., AUC) as inferred in Tahmassebi et al. (2018). Classifiers frequently perform well on one performance metric but poorly on another. For example, Decision trees and SVM classifiers achieve good performances on classification accuracy, while they yield poor performances on root mean square error as shown in Huang et al. (2003). Let's discuss some specific evaluation metrics that can be used to assess the proposed algorithm, by comparing it with state-of-the-art classification methods.

2.7.1 Cross-Validation

An over-fitted model is a common problem in machine learning. Fang (2020) demonstrate that most of the machine learning algorithms suffer from overfitting in their review, and they look at how to identify the sources of the problem and how to move this important research forward. An effective way of verifying that the model is not overfitted is cross validation. Machine learning models are evaluated by cross-validation, i.e. training them on a subset of the input data and then evaluating them on the rest. In a nutshell, we keep a portion of the data aside, and then train the model using the remaining data. A portion of the data that was held aside is then tested and evaluated for performance. By doing this, it is possible to estimate how the model will perform when it is used to make predictions using data not used for the model's training.

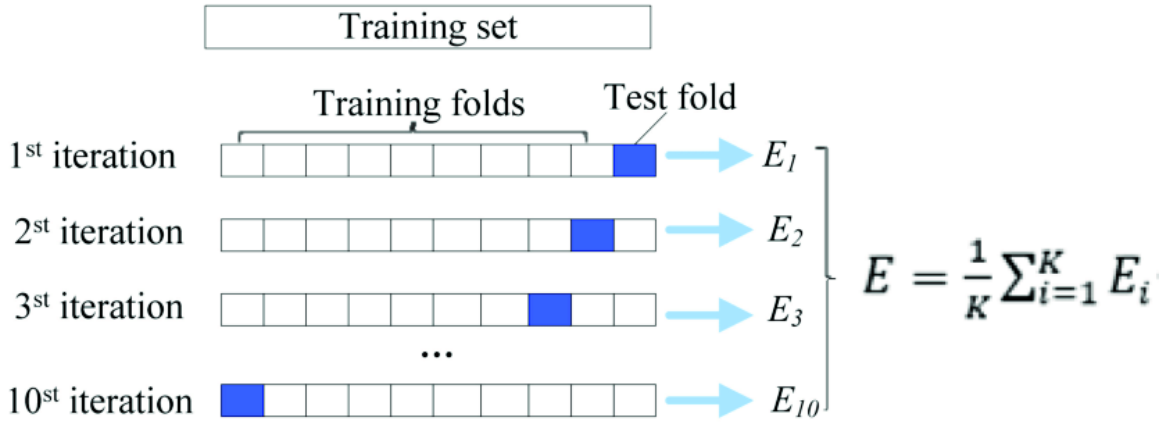


Figure 2.1: Ten-fold-cross-validation Niu et al. (2018)

In the figure 2.1, you can see that the procedure has a parameter called k that specifies how many groups a given data sample must be split into. Therefore, it is commonly called k-fold cross-validation. Ten-fold cross validation diagram. The dataset was divided into ten parts, and nine of them were used as training data, and one was used for testing. Using the average value E of the ten-group test results, an estimate of the model's accuracy is calculated and used as an indicator of the model's performance. The cross-validation error is E i in the *ith* group.3.4. As concluded by Wong (2015) a cross-validation procedure to have such sampling distributions is discussed, in order to provide new insights into evaluating the performance of classification algorithms.

2.7.2 Other evaluation metrics

When it comes to pattern recognition, information retrieval and classification (machine learning), precision (also called positive predictive value) is the proportion of relevant instances that are retrieved, while recall (also known as sensitivity) is the proportion of relevant instances that were retrieved. Tharwat (2020) explains that each of these factors will have significance depending on the purpose of the classifier and the motivation for choosing a metric. In this way, we are evaluating the performance of a model rather than just its accuracy. Below is a list of the metrics used to demonstrate the efficiency and effictiveness of

the new extensions of the co-training algorithms to the states of the art methods

- Precision - It is the ratio of positive samples correctly classified to the total number of Positive samples classified (correctly or incorrectly). The precision measures the model's accuracy in classifying a sample as positive. The precision indicates how reliable the model is at classifying samples as Positive.

$$Precision = \frac{\sum}{TP}\sum TP + FP \tag{2.1}$$

  where

  - TP True positives;
  - FP False positives.

- Recall - Recall is the ratio between the number of positive samples that were correctly classified as Positive and the total number of Positive samples. The recall reflects how well the model detects Positive samples. A higher recall means that more positive samples will be detected.

$$Recall = \frac{\sum TP}{\sum TP + FN} \tag{2.2}$$

  where

  - TP True positives;
  - FN False negatives.

- Confusion matrix - Confusion matrix is a summary of predictions on a classification problem. A list of accurate predictions and incorrect predictions for each class is provided along with a count value. This is the key to the confusion matrix. The confusion matrix illustrates the ways in which your classification model is giving incorrect predictions. Besides revealing the errors your classifier makes, it also tells you the type of error it is making.

- Hamming Loss - Based on what was explained in chapter 4 and what the authors concluded, in Wu and Zhu (2020), we will use hamming loss as part of the metric to evaluate the method. As it calculates the number of incorrect labels in relation to the total number of labels, it offers an effective measure for multi-label classification. Loss ratios are inversely proportional to the quality of the classifier. It is calculated as:

$$loss = \frac{1}{|N| \cdot |L|} \sum_{i=1}^{|N|} \sum_{j=1}^{|L|} xor\left(y_{i,j}, z_{i,j}\right) \tag{2.3}$$

  where

  - N s the total number of data samples;
  - L the total number of available classes;
  - $y_{i,j}$ is the ground truth;
  - $z_{i,j}$ is the prediction.

- Root Mean Square Error- For recommender systems, accuracy and coverage of the models is evaluated. For accuracy, the Root Mean Square (RMSE) is a standard metric used for evaluating the accuracy of predicted ratings against actual ratings. It is calculated as:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (z_i - y_i)}{N}} \tag{2.4}$$

  where

  - N s the total number of data samples;
  - y_i is the ground truth;
  - z_i is the prediction.

- ROC curve - The receiver operating characteristic (ROC) is the model selection metric for a multi-label classification problem. It is a probability curve for different classes

and it displays how good the model is for distinguishing the given classes, in terms of the predicted probability.

- McNemar Test -McNemar test is a paired non-parametric statistical hypothesis test. It can detect if the difference between the predicted values of the two classifiers is minor or significant as stated in Dietterich (1998).

- Average precision score - To evaluate the effectiveness of the recommended values. It is the average of the maximum precisions at different recall values Davis and Goadrich (2006); Everingham et al. (2010).

$$AP = \sum_{n} (R_n - R_{n-1}) P_n \tag{2.5}$$

where P_n and R_n are the precision and recall at the n-th threshold.

- F1 score - Harmonic mean of precision and recall for a more balanced summarization of model performance.

$$F1 = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{2.6}$$

2.8 Conclusions

The mining of large data sets can provide a variety of useful information if done correctly. The challenge is to figure out what information is important to provide insight from, and then determine what is important. Thus, information retrieval or data mining techniques have been demonstrated to identify categories in collected data and identify where they should be categorized. In addition, machine learning is being used in a wide range of applications, from predicting customer behavior to designing the operating system for self-driving cars.

In this chapter, we have seen how machines can be trained to analyze large datasets using unsupervised and supervised machine learning approaches. In cases in which there is less and clearly labeled data, Supervised Learning is the better choice. Unsupervised Learn-

ing would produce better performances and results when categorizing large datasets. Deep learning techniques are easier to use if the dataset is large and complex. Machine learning can enable us to uncover hidden patterns and their relationships in datasets. Machine learning algorithms help scientists identify trends over time by collecting and correlating data with their relationships and patterns, and helping teams create more effective and efficient methods of improving learning models. It is crucial for data to be of good quality, unbiased, and inclusive for machine learning to be successful otherwise it will lead to misinformation and incorrect predictions that could corrupt the data and it is very time-consuming to correct the errors that caused them. Occasionally, they might need to wait for the data to be collected in order for the algorithms to learn and develop so that they can attain significant levels of accuracy and relevance when analyzing the data. Also, it requires massive amounts of resources and requires a high level of computer power to ensure uninterrupted processing, which is extremely expensive. Even if all this is accomplished, the dataset is susceptible to being distorted by noise and errors. Furthermore, the training algorithm must recognize these as well in order to continue learning without being impacted. Despite the availability of cheap unlabeled data, most application domains lack sufficient labeled data. The process of obtaining labeled instances is very complicated since domain experts are needed to tag data patterns that are unlabeled. Co-training is a semi-supervised learning approach that can address this problem and meet the requirements of both supervised and unsupervised learning. We hope to solve the problem of inadequate labeled data during the learning stage by mixing cheap and unlabeled data with minimally labeled data.

In the chapters to come, this thesis will explore how to enhance the capacity of co-training with its approach of feature splitting, by extending it to categorize into multiple classes, as well as using collaborative filtering to recommend similarity among them. It aims to develop a cohesive framework that uses all of these features at various intervals for co-training with the objective of improving the efficiency and accuracy of the predictions as well as ensuring that the actual learning process is not too computationally intensive.

Consequently, moving to a deep learning model for analyzing even more complex and large datasets with this extended co-training algorithm.

In this thesis, the topic of study will be identifying human rights violations by analyzing a dataset of survivor stories. For the framework to be deployable across different domains, this thesis also develops a domain-free approach. For some chapters, a real-world dataset is used to demonstrate the ability to classify other datasets as well.

Chapter 3

Understanding Co-training

3.1 Introduction

As seen in the previous chapter it was noted that that in some case supervised learning or unsupervised learning on its own may not sufficient enough to classify a large unlabeled corpus. The previous chapter pointed out that supervised learning and unsupervised learning may not be sufficient in some cases to classify a large, unlabeled corpus on their own. Using learning algorithms that combine labeled and unlabeled data may improve classification efficiency. Among the methods that have attracted some attention are Expectation-Maximization (EM) Nigam et al. (2000), discriminative classification based on a generative model built from unlabeled data Jaakkola and Haussler (1999) and performance optimization for support vector machines based on transductive inference Joachims et al. (1999). According to each of these results, using unlabeled training data can significantly reduce the error rate of classification, especially if there are sparse labeled training data available.

The thesis focuses on Blum and Mitchell (1998) algorithm with labeled and unlabeled data, which is used for problem domains where features naturally divide into two sets. This algorithm classifies web pages using two classifiers: one over the words that appear on the page, and another over the words appearing in hyperlinks pointing to that page. Co-training involves datasets and algorithms that naturally partition features into two sets. Blum and

Mitchell (1998) demonstrate that learning with labeled and unlabeled data can be met if these assumptions as satisfied. They are:

- Each set of features is sufficient for classification
- The two feature sets are conditionally independent given the class

A description of the Co-training algorithm and the reasons for its success is presented in this chapter. Moreover, we will discuss the impact of assumptions on co-training algorithms in this chapter. Several examples are provided in this chapter in which co-training outperforms EM, random splitting of features, and other methods that combine labeled and unlabeled data. Last but not least, this chapter concludes with suggestions for improving the co-training. These suggestions will be helpful in shaping the next chapters in order to align with the thesis's objectives.

3.2 Overview of the Co-training Algorithm

As described in Blum and Mitchell (1998), co-training is based on the following assumptions:

- it is possible to divide the features into two categories;
- the sub-feature sets are sufficient for a good classification algorithm to be trained;
- there is a conditionally independent relationship between the two sets given that the class is the same.

In the first and second case, the views are believed to be sufficient; each view is sufficient to predict the class perfectly. This assumption is known as the sufficiency assumption. According to the third assumption, the two views must be conditionally independent. This assumption is known as the independence assumption. The sufficiency and independence assumptions guarantee co-training's success if both assumptions are satisfied. A standard

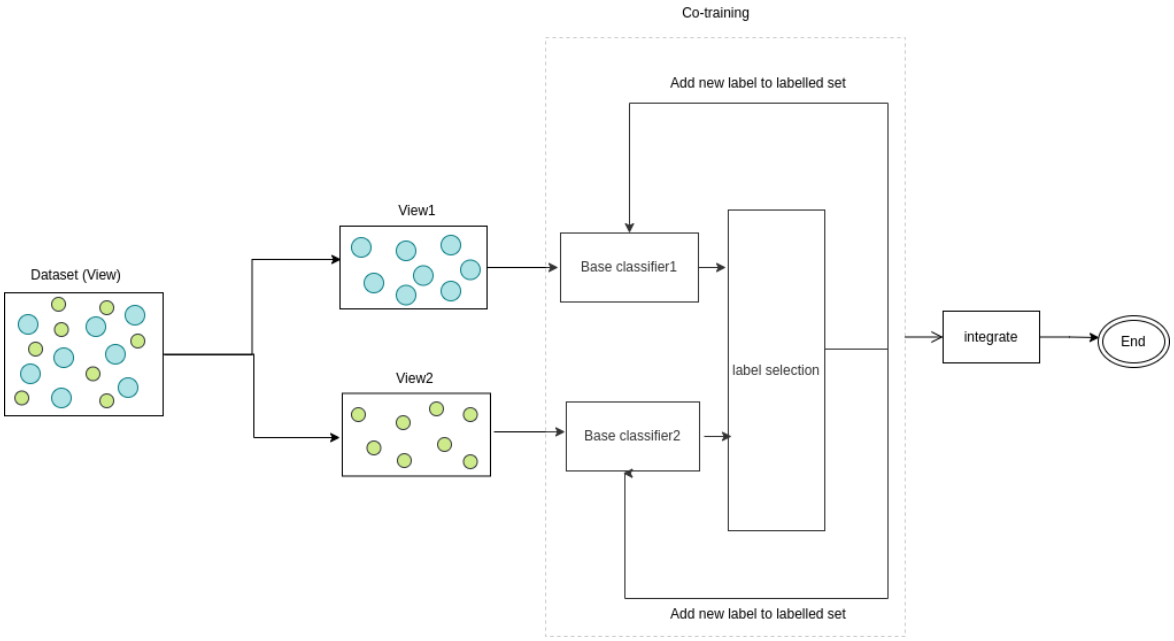


Figure 3.1: Working of the standard co-training algorithm

algorithm for co-training under two views is illustrated in Figure 3.1 using co-training as an example. The whole dataset is represented as View. The first step is to split the View using the labeled data, so that View1 and View2 can be obtained. Next, different base classifiers C1, C2 are trained using different view data, which can be used as initial classifiers. A final step involves estimating the confidence level of the unlabelled samples with the initial classifiers, and then adding the trusted samples to the training dataset for iterative training. After all the unlabelled samples have been self-labeled, the training is complete.

Initially two separate classifiers are trained with the labelled data, on the two sub-feature sets respectively. Each classifier then classifies the unlabelled data, and teaches the other classifier with the few unlabelled samples (and the predicted labels) they regard most confident. Each classifier is retrained with the additional training samples given by the other classifier, and the process repeats. In co-training, unlabelled data helps by reducing the version space size. In other words, the two classifiers (or hypotheses) must agree on the much larger unlabelled data as well as the labelled data. 1 shows the standard version of the co-training algorithm. First, a random subset U' of unlabelled samples is selected from

data set U. An initial, small training set L is used for labeling samples in U. There will be k iterations on the following steps. They are :

- Two separate classifiers, h_1 and h_2, one for each view of the dataset x, x_1 as view1 and x_2 as view2.
- For each classifier, the p samples of the positive class and the n samples of the negative class that are labelled with the highest confidence among the ones of U.
- New labels are added to L.
- Then the selected $2p+2n$ samples are then removed from U', and other $2p+2n$ samples are randomly selected from U and added to U'.

Given:
a set, L of labeled training samples
a set U of unlabeled samples
Create a pool U' of samples by choosing u samples at random from U
for *k iterations:* **do**
  Use L to train a classifier h_1 that considers only the x_1 portion of x
  Use L to train a classifier h_2 that considers only the x_2 portion of x
  Allow h_1 to label p positive and n negative samples from U'
  Allow h_2 to label p positive and n negative samples from U' ||Add these self-labeled samples to L
  Randomly choose $2p+2n$ samples from U to replenish U'
end

Algorithm 1: Blum's Co-training Algorithm

3.3 How does Co-training work?

There are many factors that affect how well the co-training algorithm works. As we proceed through this section, we are going to discuss the most important ones and explain why they are important to the algorithm's performance.

3.3.1 Creation of the views

Blum and Mitchell (1998) co-training algorithm expresses that each instance can be described using two sets of features. Classifiers are trained by sampling unlabelled data and beginning with a weak predictor in order to learn from it. It assumes that both feature sets are conditionally independent and that the target function over each of these feature sets will predict the same label. As a result of these assumptions, the paper emphasizes that the algorithm would perform exceptionally well if the instance is able to be split into two disjoint views. Feature splits and the views are a key factor which determines the performance of the co-training.

3.3.2 Sufficient and redundant feature

Using both sufficient and redundant feature subsets, which is referred to as two views co-training, establishes the diversity for co-training. A naive Bayes (NB)-based learning algorithm was used in Blum and Mitchell (1998), and page-based feature subsets and hyperlink-based feature subsets were used for the two views of the data set. Consequently, two different classifiers were built during co-training, one using page-based NBs and one using hyperlink-based NBs. Based on posterior probability, a combined classifier was generated.

3.3.3 Random feature splitting

Diversity is offered by random feature splitting in co-training. Co-training was performed on the email classification Chan et al. (2004), for example, by randomly splitting feature sets based on the classification criteria. In the experiment, the results were comparable or better than those obtained by using the original feature and natural feature partitions.

3.3.4 Underlying learning algorithms

Because the base learning algorithm differs in co-training, different learning algorithms can be used. This integration of the feature sets into the learning process is described in Goldman and Zhou (2000); Zhou and Goldman (2004), as it integrates the feature sets into the learning process. Co-training method inWang and Zhou (2007), establishes that the performance of the model can be enhanced by using two different basic learning algorithms on the two feature views.

3.4 Co-training Problems

The main challenge in co-training is ensuring that the unlabeled items are accurately labeled by the co-trainers. Starting with a small number of labeled instances, Blum's co-training involves inducing a number of classifiers to label unlabeled instances iteratively. In each iteration, multiple classifiers label unlabeled instances. Labeling the instances updates the training dataset and the classifiers are rerun over the remaining unlabeled set to improve classification accuracy. The co-training process produces two outputs: the final classification model and the combined training set from labeled and unlabeled samples. To determine which iteration is to be used, the classification model and training data need to be updated. Co-training's performance declines after a certain number of iterations, so training data must be thoroughly analyzed. They suggest that they cannot further improve the performance of the classifiers because the disagreement error rates between them will eventually converge. A higher percentage of unlabeled instances may be incorrectly labeled with each iteration of the classifier as described in Wang and Zhou (2017). Moreover, there is also the risk of over-fitting, which reduces performance.

Researchers investigated whether early stopping or reducing the number of iterations could reduce the risk of performance degradation. For the determination of the optimal number of iterations, some of the methods that were investigated include stopping the co-

training process after a certain number of iterations Kiritchenko and Matwin (2001); Wei and Keogh (2006), once there are no more unlabeled instances left to label Da Costa et al. (2019); Xu et al. (2013), and based on the performance of the classifier on a separate validation **?**Raskutti et al. (2002).

It was observed that there is no mandatory requirement in the co-training algorithm that hypothesis functions should be compatible with unlabeled data. Researchers introduced CoBoost, an algorithm that minimizes mismatch between views of the unlabeled data by combining co-training and AdaBoost Freund and Schapire (1997). An empirical investigation of the conditional independence of views underpinning co-training was performed by Nigam and Ghani (2000). Co-training performance is indeed affected by view independence, according to their experiments; however, CT is still effective when compared to other algorithms based on labeled and unlabeled data, including Expected Maximization Dempster et al. (1977), even when an explicit feature split is unknown, provided there is sufficient implicit redundancy in the data.

3.5 Conclusion

This chapter has demonstrated that co-training algorithms have gained popularity in recent years. Co-training performs differently depending on what learning method is used, we found. Co-training is an effective method for learning from small amounts of labeled data, and labeled data quality plays a crucial role in co-training. Furthermore, we highlighted the weaknesses and vulnerabilities of co-training as well as various research methods that have been investigated for improving specific predefined iterations, early stopping, and boost-stacking, etc. Following is a summary of some of the problems with the co-training algorithm.

- Without professional knowledge, it can be difficult to manually divide a single view of the current data into multiple views.

- Before selecting learners, we need to measure the difference between two learners and select those with the largest differences.
- After a certain level of training, the differences between several learners tend to diminish, making it difficult to use them for further training.
- Learners tend to attach incorrect labels to some unlabelled data at the start of iterative training, which results in a deterioration in model generalization ability.

Several methods and approaches will be discussed in the next chapters to address some of the problems associated with co-training along with improving the efficiency of multiple classifiers. The first goal of our research is to apply the co-training method to multi-label data in order to improve its performance. Using the average confidence difference, we can label the unlabeled set based on the absolute value of the differences between classes. To further enhance co-training, we will evaluate how matrix classification methods can be applied for rating labels. By using recommendation techniques, we will solve the cold-start problem by classifying similar clusters of unlabeled data through co-training. Finally, enhance co-training for deep learning by adding noise to indicate that it is noise, so the model won't be impacted by it.Our aim is to develop a theory about co-training along with empirical studies to help us identify when it is appropriate and how to improve it.

Chapter 4

Multi-label co-training

We have seen a substantial increase in the availability and collection of data across many fields as a result of the Internet and social media. This enables machine learning to improve and advance the approaches to text analysis and classification.As seen in Wei et al. (2018), creating densely labeled datasets from manually annotated texts poses a challenge due to the need for domain expertise. Data that has been labeled tend to be significantly more expensive than data that has not been. The previous chapter discussed an approach to address the problem for dealing with text classification tasks on a dataset with few labeled samples: semi-supervised learning. According to Zheng et al. (2014), in semi-supervised learning a prediction problem is addressed with only a small number of labeled training data, by utilizing both the labeled and unlabeled training data. The labeled data will provide information about the joint distribution of samples and labels, whereas the unlabeled data will provide information about the distribution of samples as shown Van Engelen and Hoos (2020).

As reviewed in Khan et al. (2010) we can see that several approaches have been developed and applied in the field of text analysis, but we are going to focus in this chapter on extending Blum and Mitchell (1998) Co-training method. Essentially, co-training consists of applying two supervised classifiers iteratively on the currently available labels, and including the predicted samples with high confidence scores into the updated set of labels. With each iteration, the training set is enlarged to incorporate the labels predicted by the classi-

fiers on the unlabeled data. Blum and Mitchell have successfully applied the Co-training algorithm for the binary classification task, wherein samples are associated with a single label. But the extended co-training method that we will discuss in this chapter assigns one story to one or more labels.

4.1 Introduction

In the traditional single-label classification methodology Tsoumakas et al. (2009), samples are assigned a single label l from a known finite set of mutually exclusive labels L. However, there are some real classification problems in which an sample can belong to more than one class simultaneously. This type of problem is known as multi-label classification Dharmadhikari et al. (2012). The main difference is that samples are associated with a set of labels Y, where $Y \subseteq L$. While multi-label learning has been applied successfully in many domains in recent years, it requires sufficient samples of labeled data to train, because there are so many possible sets of labels. It is possible to use unlabeled samples in conjunction with scarce labeled samples and help with the large labeled data requirement through co-training. However, combining co-training with multi-label learning is challenging. The challenge involves two issues: (i) a widely-known class-imbalance problem in multi-label learning Zhang et al. (2020); and (ii) determining the samples with confidence and communicating their predicted labels among classifiers for model refinement Alvarsson et al. (2021). In order to address these challenges, this chapter introduces a measure of predictive reliability to select samples and applies label-wise filtering to communicate labels confidently among co-training classifiers. In order to help practitioners who deal with human rights abuses, we have outlined an approach aimed at improving the efficiency and accuracy of categorizing texts of human rights violations/abuses. In particular, the approach is as follows:

- Establishing logical relationships among a variety of human rights abuses by creating a small labelled dataset. There are 10 categories that can be used to classify the

document set or unstructured data.

- By using Co-training semi-supervised learning for classification, two classifiers are trained over the small labelled set (as above) and they iteratively update the labelled set with new unlabelled samples classified with very high confidence by one of the two classifiers.

- Evaluate the co-trained classifier with supervised learning approaches for instance Support Vector Machine (SVM) and Logistic Regression as a baseline where the supervised learning approaches have the most of the training set labeled.

4.2 Multi-label co-training learning model

As discussed in Gokhale and Fasli (2017a), one way of achieving multi-label classification and taking advantage of labeled and unlabeled instances of multiple feature views is to integrate multi-label learning with the well-established Blum's co-training paradigm. The optimal label is determined by mutual communication among classifiers, which are individually trained on their own views of the data, thereby adding increased confidence to the labeled training set. In the next step, the respective classifiers are retrained on augmented training sets. They communicate and update until convergence occurs. In multi-label datasets, the number of samples relevant to a label is generally smaller than the number of samples irrelevant to the label. Concerns with resolving the multi-label classification problem include the class-imbalance problem and selecting an appropriate amount of samples to make a confident prediction. Let's understand why these concerns are critical to making reliable predictions that would improve algorithm performance.

- Multi-label learning has a class-imbalance problem. The number of samples relevant to a label is usually much smaller than the number of samples irrelevant to that label. As a result, the number of relevant samples can differ significantly across labels Zhang et al. (2020). Because fewer data are available to train on, there is a greater possibility

of misclassification. Furthermore, when learners communicate labels among themselves during the iterative process of co-training, the class-imbalance problem can be aggravated.

- What strategies to use to select samples for co-training and to communicate predicted labels among co-trainers with confidence. Classifiers should know how to select samples wisely and how to communicate their predictions with confidence. Due to the fact that samples to be communicated can have multiple labels, contrary to traditional co-training.

We introduce a method of multi-label classification based on co-training to address these issues. To make predictions on unlabeled samples, this co-training method trains independently on different views. The co-occurrence information of labels is then used to adjust the predicted likelihoods and deal with the class imbalance issue. Next, it sums the adjusted likelihoods across views and calculates the predictive confidence for samples based on the summarised likelihoods. Following this, it selects samples with the highest confidence, filters the likelihoods of the selected samples label-wise, and interacts with learners during the iterative co-training process. This process is repeated iteratively until convergence is reached, and then a majority vote is used to combine the predictions of the classifiers to make the final prediction.

4.2.1 Addressing class imbalance

A class imbalance occurs when there is a significant difference in the number of observations belonging to different classes. According to Zhang et al. (2020), in multi-label learning, relevant samples are much fewer in number than irrelevant samples. While cost-sensitive learning citefernandez2018cost, Synthetic Minority Oversampling Technique (SMOTE) Elreedy and Atiya (2019), and label propagation = Li et al. (2018) are all useful tools for mitigating the class imbalance. This study focused on random sampling methods as an alternative to overcoming class imbalance. This study proposed that Random Sampling would

be deployed in order to mitigate the class imbalance issue. The following are two methods for random resampling Hordri et al. (2018):

- Oversampling Duplicating samples from the minority class
- Undersampling Deleting samples from the majority class.

In other words, both oversampling and undersampling result in selecting more samples from one class than another in order to compensate for an imbalance that is already present in the data or likely to develop if a purely random sample were selected.

4.2.2 Addressing prediction confidence in samples

Communication with confidence samples and labels among classifiers during iterative processing is crucial to co-training. In traditional co-training methods, samples with the most confident predictions are directly selected for communication and model refinement Zhou and Li (2005). As for co-training with multi-label samples, since a sample may be annotated with multiple correlated labels, how to communicate labels with confidence is more challenging. This can be addressed by calculating the confidence interval that determines the score for classification error and accuracy. The accuracy of a prediction increases when the confidence interval is lower. In the case of classification error, the radius of the interval can be calculated as:

$$interval = z * \sqrt{\frac{(error * (1 - error))}{n}} \tag{4.1}$$

where

- z is the is the number of standard deviations from the Gaussian distribution
- n is the size of the sample

In the case of classification accuracy, the radius of the interval can be calculated as:

$$interval = z * \sqrt{\frac{(accuracy * (1 - accuracy))}{n}} \tag{4.2}$$

where

- z is the is the number of standard deviations from the Gaussian distribution
- n is the size of the sample

The co-training algorithm Blum is described in 1. For it, given a set L of labeled samples and a set U of unlabeled samples, the algorithm creates a smaller pool U' containing u unlabeled samples. It then iterates the following procedure. First, use L to train two distinct classifiers: $h1$ and h_2. h_1 is a naive Bayes classifier based only on the x_1 portion of the instance, and h_2 is a naive Bayes classifier based only on the x_2 portion. Second, allow each of these two classifiers to examine the unlabeled set U' and select the p samples it most confidently labels as positive, and then n samples it most confidently labels negative. They used $p = 1$ and $n = 3$. Each sample selected in this way is added to L, along with the label assigned by the classifier that selected it. Finally, the pool $U]$ is replenished by drawing $2p + 2n$ samples after running it for k iterations.

Given:
a set L and U_0 of labelled and unlabeled samples
Divide dataset D into v_1 view
k is the number of iterations
for $i \leftarrow 0$ to k by 1 do
    Generate random samples to ensure enough labels to be trained on
    Create U' by randomly selecting u samples from U_0
    Use L to train h_1 on the x_1 portion of D
    Use L to train h_2 on the x_2 portion of D
    Calculate the confidence score of predicted likelihoods for h_1 and h_2
    Select labels with highest confidence
    Update L with these labels
    Replenish U'
end

Algorithm 2: Proposed multi-label co-training Algorithm

As seen in new multi-label co-training algorithm 2 using the random sampling algorithm, additional samples are generated to calculate additional labels on the available labels. Then it randomly selects u unlabeled samples from U' and creates U', and independently

predicts the likelihood of labeling these samples in each view. It then summarizes adjusted likelihoods and overall predictive reliability of these samples in each view, and chooses the samples with the greatest degree of confidence. After this, it appends the most confident label sets obtained from the respective views to the labeled L, and removes it from the unlabeled set U' . The algorithm continues until the maximum number of iterations are reached or U' completely labeled.

4.3 Experimental Results

4.3.1 Setup

This list of ten torture mechanisms was compiled by a human rights expert. An labelled dataset was created by extracting 10 different torture mechanisms and describing the content for each. For example - Hooding with sandbags/cement bags, Blackened goggles, Plastic blindfold, Sight deprivation by other means are types of Sensory Deprivation. In this, hooding with sandbags is the content and Sensory deprivation is the label. The unlabelled set is a 1.5 GB collection of 158 testimonies and survivor stories scraped from the internet. This data was pre-processed in order to have meaningful content for classification. The normalisation of this data includes removal of all the html/css/link tags, removal of stop words and punctuation and all text being converted into lower case. The co-training classfier was built with LinearSVC as the base classifier. This is because LinearSVC is built to support multi-label classification. It is very similar to SVM with parameter kernel=linear, but implemented in terms of liblinear Fan et al. (2008) as compared to libsvm Chang and Lin (2011), which means LinearSVC minimizes the squared hinge loss, rather than just hinge loss. Furthermore, it penalizes the size of the bias (which is not SVM) so it has more flexibility in the choice of penalties and loss functions and should scale better to large numbers of samples. Such type of classification would work effectively with co-training in order to address the problem this chapter is trying to solve and assist classifying unstructured text into multiple

Label	Content
Sensory Deprivation	Blackened goggles
Sensory Deprivation	Prolonged solitary confinement with no daylight
Sensory Deprivation	Forced silence/ duct tape on mouth
Sensory Deprivation	Hooding with sandbags/cement bags
Deprivation/restrictions	Water deprivation/restriction
Deprivation/restrictions	No or inadequate bedding
Deprivation/restrictions	Food deprivation/restriction
Stress Techniques	Ski position
Stress Techniques	Prolonged squatting
Stress Techniques	Prolonged standing with arms lifted
Stress Techniques	Prolonged sitting in required position
Forced Exertion	Running/walking (including on rough surfaces) barefooted
Forced Exertion	Forced work
Temperature manipulation	Use of air-conditioning systems
Temperature manipulation	Detention in unbearably hot locations (containers, buildings with no ventilation etc.)
Sensory bombardment/use of noise	Loud radio/DVDs
Sensory bombardment/use of noise	White noise from radio/other noise/use of loud music

Figure 4.1: Labelled data set for training the classifiers

classes.

4.3.2 Baseline

A completely supervised baseline that uses 100% of the training set is trained using an SVM classifier. Results for this classifier were averaged over ten different runs with random splits of the training set. Since there is no gold standard to compare the classification results to, this is used as the baseline to be compared with a co-training and a LinearSVC classifier.

Figure 4.1 shows the labelled dataset where each torture type (label) describes the contents it considers under that type. These are extracted by querying the experimental ontology created for human rights abuses/violations.

4.3.3 Evaluation Measures

The multi-label classification problem described in this chapter is a classification problem, where each classifier tries to correctly classify the given text in to one or more of the 10 labels. The following measures have been used for the classifiers' evaluation:

- Confusion matrix to help pinpoint opportunities for improving the accuracy of the system;

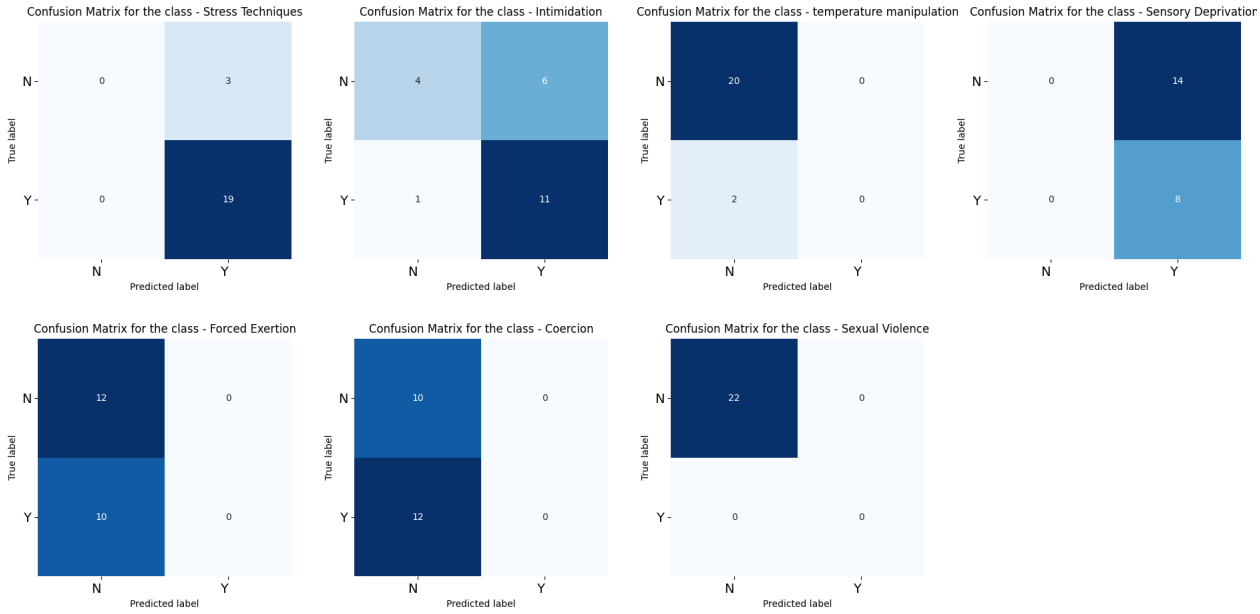


Figure 4.2: Confusion matrix for co-training classifier

- Precision and Recall curve to help point which classifier efficiently classifies all the labels correctly;
- Accuracy as its a direct measure between different classifiers based on the predicted labels.

4.3.4 Results

Each classifier is run for ten times with different splits. During each split the weighted, macro and micro average precision, recall and F1-score averaged over all different runs made during the experiments. A Count vectorizer is used for each classifier to convert the collection of training set and test set to a matrix of token counts for the classifiers to predict the classes.

Figures 4.2, 4.3, and 4.4 show the multilabel confusion matrix for the three classifiers: SVM(kernel=linear) baseline LinearSVC and co-training. The multilabel confusion matrix calculates class-wise or sample-wise multilabel confusion matrices, and in multi-class tasks, labels are binarized under a one-versus-all way. This matrix displays the percentage of true

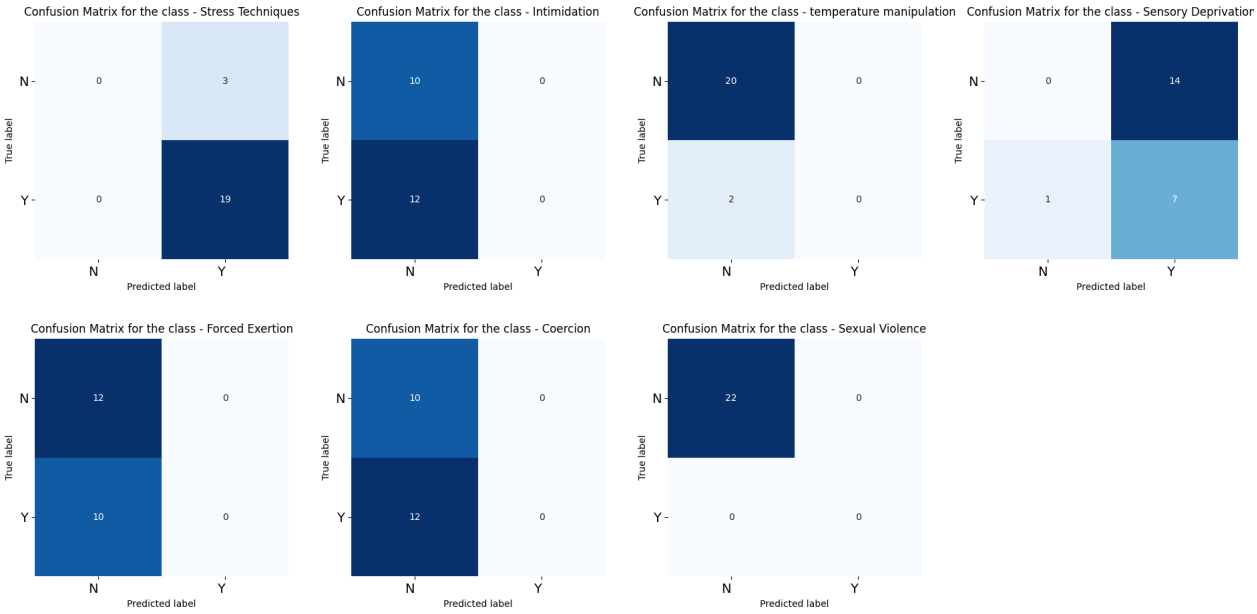


Figure 4.3: Confusion matrix for LinearSVC

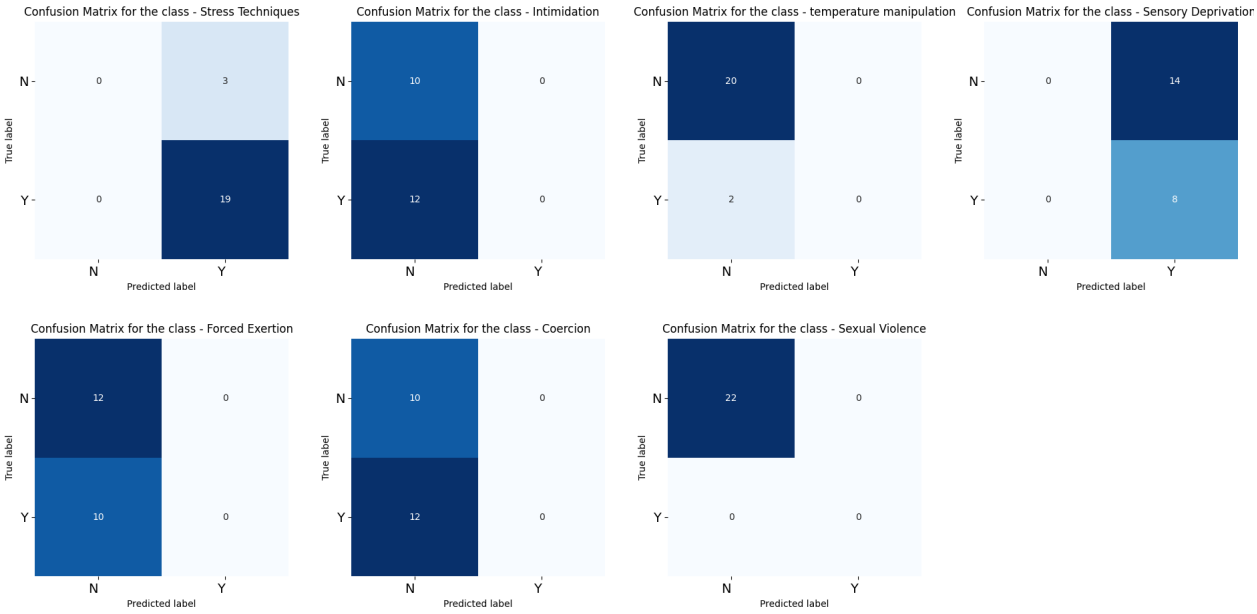


Figure 4.4: Confusion matrix for SVM(kernel=linear)

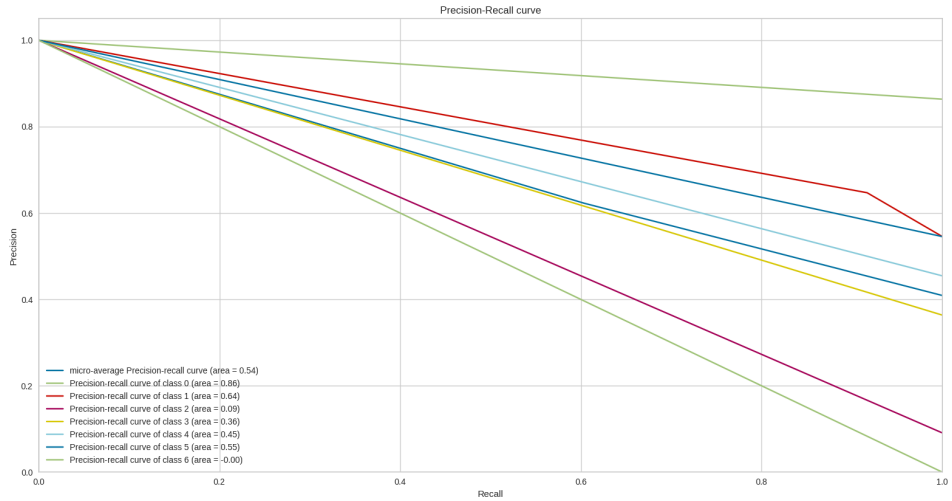


Figure 4.5: Precision/recall curve for co-training classifier

values per each label for all the labels. For these results, the LinearSVC uses 70% of the training set and the co-training classifier uses 10% initial training set which scaled up to 75% using the continuous learning process of co-training. The confusion matrix shown for each classifier uses gradient blue that shows the lightest blue as the least classified class and the darkest blue as highly predicted class. As seen from the figure, the co-training algorithm has the most accurate predictions for labels Stress Techniques and Intimidation, as the torture type. LinearSVC is trained on 70% of the training set whereas co-training is trained on 10% of the training set. Therefore, only a subset of the training set is used to train the co-training model. Then with the continuous learning of the classifier the labelled set based on the initial learning of 10% increases to labelling 75% of the unlabelled data. This new learnt classifier is then once again used to predict the unlabelled data and the results are shown on the confusion matrix 4.2.

Area Under Curve (AUC) and Receiver Operating Characteristic (ROC) curves are typically used in binary classification to evaluate the output of a classifier. In order to extend ROC curve and ROC area to multi-label like the one this chapter is trying to evaluate, it is necessary to binarize the output. One ROC curve can be drawn per label, but one can also

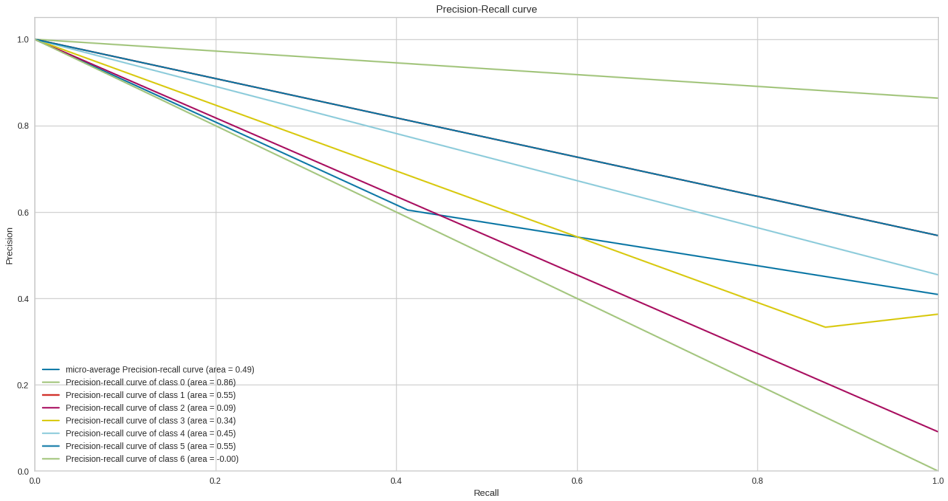


Figure 4.6: Precision/recall curve for LinearSVC

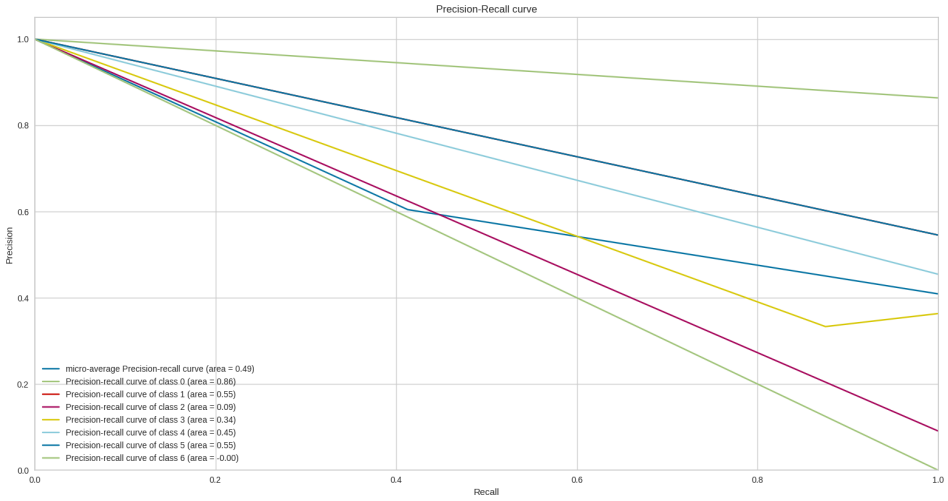


Figure 4.7: Precision/recall curve for SVM(kernel=linear)

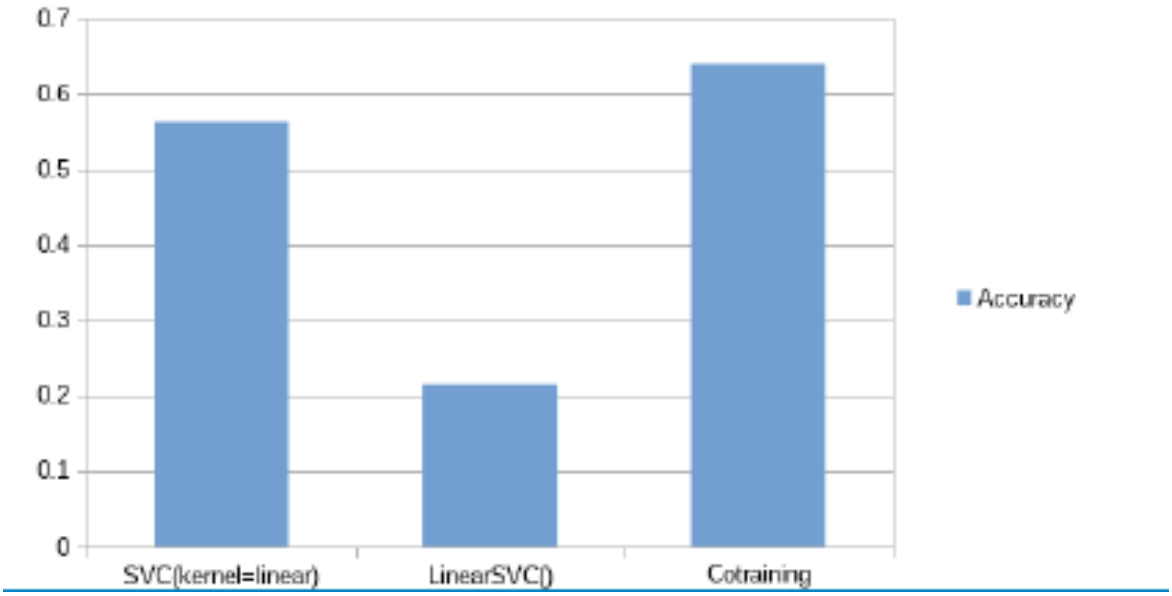


Figure 4.8: Accuracy comparison of Co-training,SVM(kernel-linear) and LinearSVC classfier

draw a ROC curve by considering each element of the label indicator matrix as a binary prediction (micro-averaging). As it can be seen in the precision and recall curve figure 4.5 the co-training classifiers yields better results, 0.44 as compared to SVM and Linear SVM. This indicates that the co-training classifier, though initially trained using just 10% of the labelled dataset performs better over a 70% trained classifiers after the continuous learning.

Figure 4.8 shows that even though the Co-training classifier was initially trained using 10% of the labelled set, it has high accuracy in predicting the correct labels. We conclude that if we train the classifier on the whole labeled set, it may make better predictions or, if trained on additional labels, it will perform better.

4.4 Conclusions and future work

Co-training classifiers classify more accurately than supervised classifiers, as demonstrated by the results. The prediction of a new label will never have a high level of confidence if there is no labelled data for the co-training algorithm to compare it to. Further testing of the

co-training algorithm is needed to find out how it does on a still larger dataset. Furthermore, it is important to ensure the consistency of performance so that the algorithm becomes a benchmark in labeling unlabelled data. We will explore this further in the next chapters using larger datasets and an extension of the co-training model.

As part of this chapter, we investigated the possibility of categorizing human rights abuses/torture based on co-training classifiers with large unlabelled datasets. Classification is based on stories shared by abuse victims and testimonies collected from them. This gave us insight into how a small labelled dataset can be helpful in classifying a large non-labelled one. It is possible for misclassification to result in wrong rehabilitation treatment and to incur losses and suffering for torture victims in the human rights domain. For this reason, an impartial approach to classifying the violations must be investigated. To do this, one must have an indication of why the classifiers predict a certain label. It may be possible to accomplish this by combining co-training with additional ranking algorithms that would establish a similarity relationship between each correct prediction. Using this idea, the following chapters will examine co-training's effectiveness.

4.5 Acknowledgement

We would like to acknowledge the support of the Economic and Social Sciences Research Council (ESRC) with grant reference ES/M010422/1 for undertaking the research described in this chapter. We would also like to acknowledge the help and support from the Human Rights Centre of the University of Essex and the charity Freedom from Torture in constructing the ontology for the domain of human rights.

Chapter 5

Matrix factorization for co-training

In the previous chapter, the co-training algorithm was extended to support multi-label classification in order to improve the performance of a semi-supervised learning model by incorporating a small amount of unlabeled data so as to increase the labeled set. In the absence of two independent and self-sustaining views of the data, applying co-training remains challenging. The purpose of this chapter is to explain how random features can be divided and that matrix factorization can be used to discover latent features that explain interactions between entities within a single view dataset. Labeled views remain consistent with co-training assumptions, but balance biased information. As a result, weak classifiers from the labeled views are combined in such a way as to enhance prediction accuracy. Any classification to try and group samples or objects into a single class tries to do this in the majority of cases. In spite of this, human rights victims can suffer a variety of different violations or abuses. This chapter focuses on all of these aspects using a semi-supervised classification model based on the effectiveness of matrix collaborative filtering to classify narratives told by victims into one or more categories of human rights abuses.

While it can be tough to obtain a sufficient amount of labeled data for any learning task, unlabeled data can be acquired comparatively easier and they are typically plentiful. Taking advantage of two conditionally independent and separate sets of labeled data, the co-training algorithm by Blum and Mitchell (1998) attempts to achieve better learning performance by

interpolating the insufficiently labeled data with large amounts of unlabeled data. This thesis explores human rights abuses and victims' stories, a field in which it is difficult, however, to separate survivor stories into such independent views, which narrows the scope of the co-training model. Abney (2002) suggests that if the independence assumption on views is discounted, then co-training would still be applicable under a weaker independence assumption and would thus be as effective as conditional independence. This method makes co-training algorithms for classification better by maximizing agreement in unlabeled data for classification. In order to prevent duplicate labels from being assigned to views, this method randomly splits data from a single view into two views, which are then assumed to be self-sufficient in terms of correct classification. Data from labeled and unlabeled datasets are decomposed into lower rank factorization by a technique called Singular Value Decomposition (SVD), which converts them into factors based on patterns in label rankings. In general, machine learning algorithms are used to investigate problems that involve single labels. The co-training model is, however, extended to a multi-label classification since the victim may have been subject to more than one violation.

For partially ranked data, recommendations based on matrix factorization (MF) as proposed by Koren et al. (2009) have achieved acceptable results for prediction ratings. By implementing low-rank matrix factorization as shown by McAuley et al. (2012a); Sarwar et al. (2002); Yu et al. (2009) a latent factor model can predict the user's ratings on previously unrated items by looking for hidden patterns between the user matrix and the item matrix. Herlocker et al. (2004) uses this method of recommendation in many domains where users do not search reviews directly but are recommended products that would best suit their preferences. Netflix and Amazon rely on ratings to make recommendations, as seen in Zhou et al. (2008) or Mangalindan (2012). The analysis of various types of beers in McAuley et al. (2012b) is another well-known study area for recommendation using matrix factorization. There are patterns that can explain why some people prefer certain types of beer and avoid others.

5.1 Method

With competition between internet-based organisations for instance Amazon, Netflix and Youtube intensifying, maintaining and increasing user satisfaction so that users can remain loyal is of paramount importance; typically exemplified through relating their preferences to the items they view, rank, shop and like. As seen in Konstan and Riedl (2012), organisations commonly use recommender systems that can analyze the trends or patterns exhibited by the user and associate the next best items for them. Such systems are based on two distinct methods:

- **Content-based Filtering -** According to Van Meteren and Van Someren (2000), a content-based approach tries to determine a user's features or behavior based on the items they are likely to respond positively to. According to content-based methods, the main idea is to construct a model to explain the observed interactions between users and items based on the "features" available. As we continue to consider users and items, the model can also provide us with an insight into why this occurred. This model allows us to predict a user's preference pretty easily by looking at their profile and deciding on their terms, based on the information they provide. This technique relies on a lot of domain knowledge since the feature representations of the items are created based on the user interactions. Therefore, the model can only be as good as its custom features. This means that it can only give suggestions based on what the user already likes. The model is therefore limited in building upon users' existing interests.

- **Collaborative Filtering -** As stated in Resnick et al. (1994), collaborative Filtering tends to find what similar users would like and the recommendations to be provided and in order to classify the users into clusters of similar types and recommend each user according to the preference of its cluster. The main idea that governs the collaborative methods is that through past user-item interactions when processed through

the system, it becomes sufficient to detect similar users or similar items to make predictions based on these estimated facts and insights. The only issue with this method is that the prediction of the model for a given user, item pair is the dot product of the corresponding embeddings. So, if an item is not seen during training, the system cannot generally create an embedding for it and hence cannot query the model with this item. This issue is known as the cold-start problem.

In this chapter we will discuss how matrix factorization is used to derive hidden and less obvious relationships between a few human rights experts' labels given for the surviving victim stories. According to the approach talked about in Gokhale and Fasli (2018) using this approach with collaborative filtering because it identifies the latent features that exist in different types of stories and determines whether they are similar or not. Based on two additional models, the nearest neighborhood model and the latent factor model, a collaborative filter can provide recommendations. In contrast to the neighborhood model, in which use of the relations between stories within the neighboring area is evaluated, this study focuses on the second approach, in which stories and labels are addressed equally. Due to the factors affecting the level of abuse the victim may experience, it is impossible to establish a direct correlation between the type of abuse the victim experiences and the label. This is because intensity, duration, and other factors affect the degree of abuse that a victim could experience. In this method, matrix factorization is used to compute additional labels based on the small labeled dataset in order to identify violations using the latent factor model. The chapter elaborates further on how matrix factorization could be used to identify labels with the highest level of confidence in co-training.

5.1.1 Matrix factorization

A basic concept of this model is that both items and users (using these stories as examples) can be distinguished by feature vectors based on patterns and trends. Furthermore, the rating of a particular item (e.g. a label) can be related to a user (story). In the current

storyid	story	Sensory Deprivation	Stress Techniques	temperature manipulation	Coercion	Intimidation	Sexual violence	Forced Exertion
1	Hooding with sandbags/cement bags	6	7	0	0	0	1	2
2	Blackened goggles	0	7	3	0	0	1	0
3	Forced silence/ duct tape on mouth	0	6	2	4	7	0	0
4	Prolonged sitting in required position	7	0	0	0	3	1	0
5	Prolonged standing/wall standing	7	5	0	4	3	1	6
6	Sexual violence to genitals	3	0	0	0	6	7	2
7	Molestation	3	4	1	0	0	7	2
8	Penetration using instruments	0	0	1	5	0	7	0
9	Pressed on hot surfaces	6	0	7	0	0	0	0
10	Stamping on victim	0	6	0	5	0	0	7
11	Dragging victim along ground	0	7	2	4	5	0	3
12	Simulated drowning	0	0	0	0	3	0	7
13	Verbal abuse	0	0	2	0	7	1	4
14	Detention in unbearably hot locations	6	3	7	0	0	0	0

Figure 5.1: Matrix of user stories labeled by human rights experts.

labeled dataset, some stories have been rated against each label, but the rest have been left empty. The goal now is to rate the remaining stories to predict ratings for new stories based on ratings from the unlabeled set. It is possible to create a vector of stories that reveals the missing data by placing the stories and their labels in a two-dimensional matrix. When this is applied to the set of sparsely labeled victims stories and the rankings associated with them, the initial matrix appears as in figure 5.1.

In figure 5.1, there are 14 unique story lines, each having 7 labels ranked from 1-10 (1-lowest to 10 -highest). Those which are marked as 0 are the ones which are not rated by the experts.

5.1.2 Ranking

Matrix factorization models map both users and items to a joint latent factor space of dimensionality lf, such that user-item (story-label) interactions are modelled as inner products in that space. Let $\mathbf{R}$ of size $|M| \times |N|$ be the matrix that contains all the ratings that the stories have assigned to the labels.Here M is the stories matrix and N is its labels matrix. The task is to discover K latent features by finding out two matrices $\mathbf{P}(|M| \times Kmatrix)$ and

storyid	story	Sensory Deprivation	Stress Techniques	temperature manipulation	Coercion	Intimidation	Sexual violence	Forced Exertion
1	Hooding with sandbags/cement bags	6	7				1	2
2	Blackened goggles		7	3			1	
3	Forced silence/ duct tape on mouth		6	2	4	7		
4	Prolonged sitting in required position	7				3	1	
5	Prolonged standing/wall standing	7	5		4	3	1	6
6	Sexual violence to genitals	3				6	7	2
7	Molestation	3	4	1			7	2
8	Penetration using instruments			1	5		7	
9	Pressed on hot surfaces	6		7				
10	Stamping on victim		6		5			7
11	Dragging victim along ground		7	2	4	5		3
12	Simulated drowning					3		7
13	Verbal abuse			2		7	1	4
14	Detention in unbearably hot locations	6	3	7				

Figure 5.2: Initial ranking of all the labels for each story id by the human right experts.

$\mathbf{Q}(|N| \times Kmatrix)$ such that their product approximates $\mathbf{R}$:

$$\mathbf{R} \approx \mathbf{P} \times \mathbf{Q}^T = \hat{\mathbf{R}} \tag{5.1}$$

In this way, each row of $\mathbf{P}$ would represent the strength of the associations between a user (story) and the features. Similarly, each row of $\mathbf{Q}$ would represent the strength of the associations between an item (label) and the features. To get the prediction of a rating of an item (label) n_j by m_i, the dot product of the two vectors corresponding to m_i and m_j is calculated:

$$\hat{r}_{ij} = p_i^T q_j = \sum_{k=1}^{k} p_{ik} q_{kj} \tag{5.2}$$

However, in the above case, the chances of over-fitting the features is high. In order to reduce over-fitting and penalize complexities (if any), λ is added in as constant to control the regularization as follows:

$$e_{ij}^2 = (r_{ij} - \sum_{k=1}^{K} p_{ik} q_{kj})^2 + \frac{\lambda}{2} \sum_{k=1}^{K} (||P||^2 + ||Q||^2) \tag{5.3}$$

This constant, controls the magnitudes of the feature vectors of U and D to obtain a good approximation. This is applied to the sparsely labeled set of victim stories and their respective rankings and is re-executed to generate $\mathbf{P}$ and $\mathbf{Q}$. Once $\mathbf{P}$ and $\mathbf{Q}$ are derived from $\mathbf{R}$, their dot product is taken to produce the predicted ratings for the missing values.

Figure 5.2 describes how each story (by its id) is categorized by ranks into 7 different

storyid	story	Sensory Deprivation	Stress Techniques	temperature manipulation	Coercion	Intimidation	Sexual violence	Forced Exertion
1	Hooding with sandbags/cement bags	5.95	6.69	8	3.76	0.77	1.19	2.4
2	Blackened goggles	9.16	6.62	3.32	5.38	8.26	1.19	7.37
3	Forced silence/ duct tape on mouth	5.84	5.89	1.88	4.3	6.91	4.39	4.44
4	Prolonged sitting in required position	7	6	4.92	4.2	2.97	1.01	4.66
5	Prolonged standing/wall standing	7.03	5.55	4.44	4.07	3.48	0.42	4.94
6	Sexual violence to genitals	3.03	5.18	0.51	3.35	5.94	7.03	2.02
7	Molestation	2.78	4.84	0.56	3.1	5.36	6.55	1.81
8	Penetration using instruments	5.42	6.71	1.16	4.69	8.32	7.09	4.06
9	Pressed on hot surfaces	6.09	7.39	6.87	4.32	0.2	3.36	2.77
10	Stamping on victim	9.32	6.08	5.32	4.85	4.61	0.65	7
11	Dragging victim along ground	4.69	6.44	2.44	4.15	5.24	6.33	2.92
12	Simulated drowning	9.9	6.67	6.87	5.06	2.99	0.25	6.99
13	Verbal abuse	5.83	4.24	1.46	3.54	6.37	1.32	4.89
14	Detention in unbearably hot locations	5.88	3.19	6.91	2.18	0.58	0.12	3.39

Figure 5.3: New ranks for the stories after matrix factorization.

types of violations. There are some gaps seen in the figure, this means that these labels are either not ranked or have a very low rating. Hence, the task of predicting the missing ratings can be considered as filling in the blanks so that the values would be consistent with the existing ratings in the matrix to generate a robust training set for the classifiers to be trained on.

Figure 5.3 is the output of the dot product **P** and **Q**, where **P** and **Q** are derived from the initial matrix shown in Figure 5.1. As seen in Figure 5.3, the new ranks are approximately close to those ranked by the human rights experts. Looking at this graph or the matrix it generates, similarities between stories can be ascertained. Labels with ranks greater than 4 are For example it is reasonable to say:

- Labels with a rank over 4 are considered to be labels for that story
- story ids 4 and 5 which have the story as 'Prolonged sitting in required position' and 'Prolonged standing/wall standing' respectively with the same labels 'Forced Exertion,Sensory Deprivation,Stress Techniques'
- story ids 6 and 7 are similar having content as 'Molestation' and 'Penetration using instruments' respectively with label as 'Coercion, Intimidation, Sexual violence, Stress Techniques'

This supports the hypotheses of the chapter which is that a large labeled dataset can be created using minimal input from experts and matrix factorization.

storyid	story	Sensory Deprivation	Stress Techniques	temperature manipulation	Coercion	Intimidation	Sexual violence	Forced Exertion
1	Hooding with sandbags/cement bags	3.2147273226	2.046006569	5.5909608046	5.9242696968	0.982876927	7.0320166607	0.4174157072
2	Blackened goggles	3.9442946182	3.9843324972	5.1234939492	7.1067089121	0.996651268	6.9881835064	2.9785401435
3	Forced silence/ duct tape on mouth	4.4315010247	3.0116214957	6.6229118586	5.3906535757	4.903167234	6.1074687399	1.798351025
4	Prolonged sitting in required position	4.1230720701	6.144849837	2.9588177475	7.0038962152	1.029695498	4.8486013121	6.4944344626
5	Prolonged standing/wall standing	4.062694586	5.9568889995	3.0486611288	6.9690587001	0.954610467	4.9563341857	6.2167206022
6	Sexual violence to genitals	3.9336414641	1.9951026597	6.0864629048	2.9633373227	6.909618794	3.8537430608	1.0830950093
7	Molestation	3.9755364151	1.9765476467	6.2052313594	3.0219093764	6.959483939	3.9667833608	1.0206085795
8	Penetration using instruments	4.8172638615	2.5101945139	7.7146796952	4.7115552171	7.03771962	5.9795870945	1.0811290252
9	Pressed on hot surfaces	3.9046414596	6.1884069039	2.1873373887	6.0151519615	1.671735751	3.4744483708	6.9571413002
10	Stamping on victim	4.9587779801	6.9753055137	3.9762012417	8.2406481201	1.593884081	6.0132739146	7.1912062245
11	Dragging victim along ground	3.6834220406	3.2013498232	5.3136396356	6.4279223252	1.349532151	6.7783024007	1.9994847017
12	Simulated drowning	4.454539252	6.9702152151	2.9919284357	8.1256904394	0.293018068	5.5985612881	7.3759197659
13	Verbal abuse	4.5943064796	3.8582009841	6.9450603629	8.5409190044	1.042337086	9.2272914119	2.1016451906
14	Detention in unbearably hot locations	2.9324189507	6.0065504874	0.5367366549	5.9551869684	0.997328234	3.0410517305	6.9786170995
15	Reprieve - The stories of three men executed by firing squad	2.2196904255	2.4541144773	2.5545977562	3.7223981086	0.860018441	3.3863205633	2.1139064976
16	Freedom from Torture - The true legacy of torture	2.2836297698	2.0814854755	3.058666158	3.584158564	1.329559973	3.6167726748	1.5294354538
17	Teresa Celia Meschiati	1.6702526517	0.7962468346	2.8960243096	2.0711632805	1.893346544	2.681508761	0.1101188087
18	testimonies of torture survivors	0.9252844495	0.7934193293	1.1644716144	1.0289504738	1.102283727	0.9959810888	0.6766465247
19	Tesfaye's Story	2.3425400171	3.2437163936	1.7720502541	3.3632600957	1.454982893	2.2424225213	3.4865378069
20	Saad's Story	2.7891462552	3.212030705	2.7988767054	3.8295944064	2.144390338	3.0914828881	3.1441968783

Figure 5.4: New labels obtained after applying matrix factorization to the unlabeled data.

Updating the labeled set with new ranks, a selected few unlabeled samples from the unlabeled data set are added to the training set to predict their rating in order to find recommended stories and the labels for them. As seen in figure 5.4, story 15 and below are the newly added stories which are now ranked with respect to the earlier ranked stories. Additionally it can be seen that the newly added stories with id 15 and 16 are similar and thus labeled as 'Coercion, Sexual Violence, temperature manipulation", where as story ids 19 and 20 are labeled as "Stress Techniques, Coercion, Forced Exertion'. Labels that have a ranking 3 and above out of 7 are considered good enough to be added to the initial labeled set . This process is repeated a finite number of times or until all the unlabeled data are labeled.

The advantage of using matrix factorization is that results can be inferred through using minimal input from experts (which is extremely time consuming as well as expensive). This assists in getting a larger labeled sample set to train the classifier.

5.1.3 Generating recommendations

After building the neighborhood group of relevant ratings of the missing labels in the labeled dataset, recommendations are generated for similar stories from the unlabeled set. Collaborative Filtering generates recommendations as either prediction, which is one numerical value, or recommendation, which is a list of top N labels. Since we investigate to use the latent factor model that recommends based on prior behaviour, similarities between the stories

Unlabeled story	Recommemded story	Recommended label
Survivor Stories Building Freedom Brick by Brick	Hooding with sandbags/cement bags	Sensory Deprivation;Intimidation
Survivor Stories Building Freedom Brick by Brick	Forced silence/ duct tape on mouth	Sensory Deprivation;Intimidation
Survivor Stories Building Freedom Brick by Brick	Prolonged sitting in required position	Stress Techniques;Forced Exertion
Survivor Stories Building Freedom Brick by Brick	Sexual violence to genitals	Sexual Violence;Coercion;Intimidation
Survivor Stories Building Freedom Brick by Brick	Detention in unbearably hot locations	Sensory Deprivation;temperature manipulation
Youssef's Story	Hooding with sandbags/cement bags	Sensory Deprivation;Intimidation
Youssef's Story	Forced silence/ duct tape on mouth	Sensory Deprivation;Intimidation
Youssef's Story	Prolonged sitting in required position	Stress Techniques;Forced Exertion
Youssef's Story	Sexual violence to genitals	Sexual Violence;Coercion;Intimidation
Youssef's Story	Pressed on hot surfaces	Stress Techniques;temperature manipulation;Coercion
Peter's Story	Hooding with sandbags/cement bags	Sensory Deprivation;Intimidation
Peter's Story	Forced silence/ duct tape on mouth	Sensory Deprivation;Intimidation
Peter's Story	Prolonged sitting in required position	Stress Techniques;Forced Exertion
Peter's Story	Sexual violence to genitals	Sexual Violence;Coercion;Intimidation
Peter's Story	Detention in unbearably hot locations	Sensory Deprivation;temperature manipulation
Yonas' Story	Hooding with sandbags/cement bags	Sensory Deprivation;Intimidation
Yonas' Story	Prolonged sitting in required position	Stress Techniques;Forced Exertion
Yonas' Story	Sexual violence to genitals	Sexual Violence;Coercion;Intimidation
Yonas' Story	Detention in unbearably hot locations	Sensory Deprivation;temperature manipulation
Yonas' Story	Dr. Candeloro, Husband of Marta Garcia de Candeloro.	Sensory Deprivation,Stress Techniques,temperature manipulation

Figure 5.5: Top 4 recommendations with labels for 5 randomly selected unlabeled stories

have to be determined.

Applying the cosine similarity along with (SVD) Sarwar et al. (2002), the similarities can be identified between the stories to predict the labels. The benefit of using both cosine similarity and SVD is to avoid negative correlation due to lack of data and handle the issues of scalability and sparsity.

$$cos(\boldsymbol{x},\boldsymbol{y}) = \frac{\boldsymbol{x}\cdot\boldsymbol{y}}{||\boldsymbol{x}||\cdot||\boldsymbol{y}||} \tag{5.4}$$

SVD

$$A_{mn} = U_{fv} \times S_r \times V^T \tag{5.5}$$

Matrix A_{mn} is called utility matrix, in which each row i represents the list of rated label for story i while each column j lists all the stories who have rated label j. Using SVD it can be factorized to U_{fv}, S_r and V. The U_{fv} matrix represents the feature vectors corresponding to the stories in the hidden feature space, S_r describes the strength of each latent factor and the V matrix represents the feature vectors corresponding to the labels in the hidden feature space. This calculates the vector which establishes the relationship between the stories and theirs labels. With this newly identified relationship between the stories and labels, the recommender lists top n stories similar to the story it needs recommendations for.

Figure 5.5 displays the top 5 recommendations for 4 randomly selected stories from the

unlabeled dataset. As seen from the figure, each unlabeled story has 4 recommended stories and labels based on similar features that have been discovered using matrix factorization. Each of the recommended stories are ranked based on the average of individual ranks of each label. The figure thus display the 4 stories in descending order of their ranks. Out of the 4 recommendations, the story with the highest rank will be selected as the label for the unlabeled story.

5.1.4 Co-training extension

Blum and Mitchell Blum and Mitchell (1998) proposed a conditional independence assumption to combine labeled and unlabeled data to classify the content into relevant classes. To improve and extend their algorithm, this method uses matrix factorization to rank first the missing labels. The two classifiers are trained on the newly ranked labeled set to predict the classes for the entire unlabeled dataset based on the predicted ranks for selected unlabeled samples. Throughout the entire unlabeled dataset, the trained classifiers are then applied multiple times to eventually predict their labels. Predictions are averaged over several iterations to arrive at a final prediction. Due to the random division of the training data into two separate sections, each of two classifiers is trained on different segments and this might lead either classifier to not have enough training data on which to train, resulting in incorrect results.

The algorithm developed by Blum and Mitchell only works for binary classification. Using this method, the unlabeled data can be classified into multiple labels, expanding the scope of co-training.

The proposed algorithm to deal with multiple labels is shown in Algorithm 4.

Given:
a set L of labeled training samples with only some samples ranked
Use MF rank the training samples
Create a pool U' by choosing u at random from U
Divide training set X into x1 and x2 randomly
h1 and h2 are two classifiers that are to be trained
for $i \leftarrow 0$**to**k**by**1 **do**
  Use MF rank U' samples
  Use L to train h1 on the x1 portion of x
  Use L to train h2 on the x2 portion of x
  Use R to recommend top n recommendations for U'
  Compare recommendations with prediction for U' by h1 for x1 ||Select the highest ranked prediction as positive samples
  Compare recommendations with prediction for U' by h1 for x1 portion
  Select the lowest ranked prediction as negative samples
  Compare recommendations with prediction for U' by h2 for x2
  Select the highest ranked prediction
  Compare recommendations with prediction for U' by h2 for x2
  Select the lowest ranked prediction as negative samples
  Label p positive and n negative samples from U'
  Label p positive and n negative samples from U'
  Add these self-labeled samples to L
  Randomly choose 2p + 2n samples from U to replenish U'
end

Algorithm 3: Proposed Co-training Algorithm

5.2 Experimental Setup

5.2.1 Dataset

As explained in the previous chapter, we are using the same dataset as we did earlier, but we have added a few additional stories as a result of our ongoing search for new victim stories on the internet.

The unlabeled set is a 2.8 MB collection of 238 testimonies and survivor stories scraped from the Internet. This data was preprocessed in order to have meaningful content for classification. The normalization of this data includes removal of all the HTML/css/link tags, removal of stop words and punctuation and all text being converted into lower case.

5.2.2 Performance Evaluation

5.2.3 Classifiers

For thorough comparison, the proposed co-training classification model is evaluated against state-of-the-art supervised classifiers, as well as some variants of co-training. The baselines considered are:

- **SVC** a widely used supervised text classifier. The linear kernel and onevs-rest scheme with TF-IDF weighting.
- **Decision Trees** - A popular classifier where trees are constructed where each node corresponds to a group of instances from the dataset. They can be adapted by taking multiple labels into consideration in decision functions.
- **MlKNN** - Zhang and Zhou (2005) propose a new multi-label learning algorithm based on K-Nearest Neighbours (KNN). This model uses a lazy-learning approach.
- **Logistic Regression** - In the multi-label case, Logistic Regression uses the one-vs-rest (OvR) scheme and uses the cross-entropy loss when the option is set to 'multinomial'.
- **GaussianNB** - A multi class text classifier which is used with the onevs-rest scheme provides apt multi label classification.
- **Random Forest** - A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.
- **Co-training algorithm for SVC (kernel=linear), Logistic Regression, Decision Trees, Random Forest and NaiveBayes** - Using the co-training model to classify human right abuses/violations Gokhale and Fasli (2017b).

5.2.4 Metrics

For multi-label classification, to evaluate the performance, it is necessary to evaluate the ranking prediction of the most relevant documents for each category in order to quantify the quality of the predicted values. The following measures have been used for the classifiers' evaluation:

- classifiers are evaluated for their macro-averaged F1 and hamming loss;
- For accuracy, the Root Mean Square (RMSE) is a popular state-of-the-art metric which is used for evaluating predicted ratings with actual ratings;
- The average precision score is also used to evaluate the effectiveness of the recommended values;

5.2.5 Results

Figure 5.6 provides a comparison overview of the various state of art multi-label classifiers, the co-training algorithm with label ranking and the proposed collaborative co-training algorithm. Based on the results, the chapter shows that overall the coverage error rate displayed by the improved co-training algorithm is much lower than its relevant counterparts. This indicates that the predicted labels (one or more) for a story match completely the truth value for it. The truth value for the stories was manually recorded by reading through each story one by one.

It is also seen from the results, that the macro F1 measure evaluates the ability of the algorithms to correctly identify the relevance of each label. RandomForest classifier combined with the proposed co-training algorithm provides the best results amongst all the classifiers used for the experiment. The result determines that the improved co-training RandomForest classifier has the highest chance to correctly identify the relevance of each label.

In a multi-label scenario, Hamming Loss is an approximate prediction that can be efficiently computed from label-wise information and hence determines the accuracy of the

classifier. As seen in Figure 5.6, when a GaussianNB classifier is teamed up with the proposed collaborative co-training classifier, it performs significantly better than the rest. This means that the percentage of true labels predicted matches exactly the manually recorded ground truth. This also aligns with the fact that Hamming Loss is a binary relevance method, which only trains a learner for each label without taking into account dependencies. The proposed collaborative co-training model relies on using the OneVsRest Classifier, whose approach is to fit one classifier per class. ROC curves are another good measure for determining the accuracy of any classifier. As seen in figure 5.6, KNN with the collaborative co-training model classifier performs marginally better as compared to the others. This indicates that the ability of the new recommender model along with co-training is higher than standard state-of-the-art classifiers. This also can be seen in figure 5.7, that displays the accuracy of the newly proposed collaborative co-training algorithm paired with currently efficiently working classifiers. Thus the GaussianNB classifier predicts up to 70% correct labels for stories followed by the DecisionTree classifier.

Cosine similarity is used to build the recommender as it determines the similarity between two vectors by measuring the angle between them. Lower values of RMSE help confirm that the vectors are in the neighbourhood of each other and the aim of the recommender is to minimise the RMSE while predicting the labels for the stories. An RMSE of 0.5 implies that on average, the recommender is approximately 0.5 off with each prediction. However an RMSE over 0.6 is considered to be a good recommended prediction. Thus the classifier that performs the best recommending over 74% correct predictions is the GaussianNB classifier.

When comparing the labels predicted for the stories with the ground truth, figure 5.9 shows the variations of the labels for the stories. As seen from the figure, the predictions made by the GaussianNB classifier are relatively closer to the truth value as compared to the other classifiers.

Metrics	Average precision Score	Coverage Error	Hamming Loss	F1 macro averaging	ROC
Decision Tree	33.2186	7	0.5210084034	0.1743816837	0.4907631129
Decision Tree with co-training	20	7	0.5357142857	0.1746530617	0.52417144
Decision Tree with collaborative cotraining	32.8835	5.6764705882	0.5453501401	0.3602407427	0.5872473233
GaussianNB	33.0937	6.6078431373	0.6848739496	0.3588041085	0.4821626928
GaussianNB with co-training	22.6272	7	0.4285714286	0.2932653061	0.4285714286
GaussianNB with collaborative cotraining	33.7344	5.6862745098	0.7008403361	0.2624585094	0.5019943329
KNN	33.1666	6.4607843137	0.3767507003	0.1709825063	0.4886688667
KNN with co-training	20.0256	7	0.4142857143	0.22	0.5428571429
KNN with collaborative cotraining	33.4175	6.2549019608	0.3991596639	0.2902325899	0.5994271601
Logistic regression	33.5038	6.6568627451	0.3921568627	0.1635151639	0.5031892511
Logistic regression with co-training	22.3077	7	0.4285714286	0.1632653061	0.4885714286
Logistic regression with collaborative cotraining	33.5549	6.1117647059	0.4243697479	0.2574040281	0.5042276872
Random Forest	33.3333	7	0.5182072829	0.1766843747	0.5
Random Forest with co-training	18.8462	7	0.4	0.1176470588	0.5142857143
Random Forest with collaborative cotraining	33.2733	5.7352941176	0.531372549	0.3937449168	0.5956621861
SVC	33.4348	6.6568627451	0.393557423	0.1650969168	0.4998759381
SVC with co-training	23.0769	7	0.4282156863	0.2142857143	0.5242857143
SVC with collaborative cotraining	33.3333	5.862745098	0.4291596639	0.279327904	0.5

Figure 5.6: Comparison of various state of art classification models with the proposed co-training classifier.

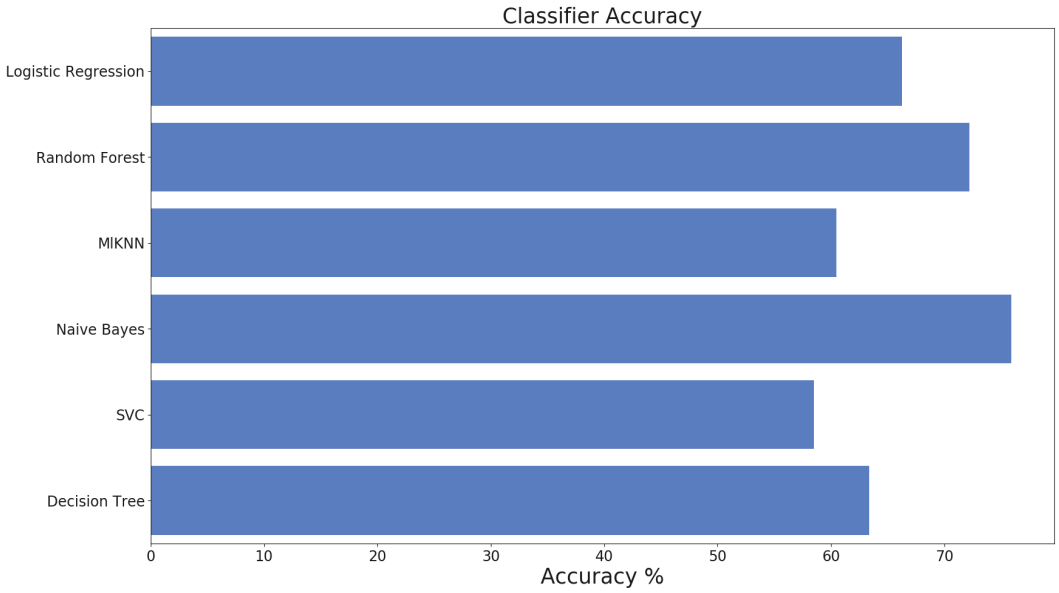


Figure 5.7: Classifier accuracy for SVC (kernel=linear), LogisticRegression, MlKNN, RandomForest, DecisionTree, GaussianNB paired up with the proposed co-training classifier.

5.3 Conclusions

In this chapter, the co-training algorithm is extended by adding multi-label classification and matrix factorization that identify underlying similarities between stories and labels. Additionally, the recommender system accurately predicts the stories that will be similar to those predicted by the new label. Hence, demonstrating that the co-training process could be extended to make use of two classifiers to achieve optimal classification speeds. As we saw in

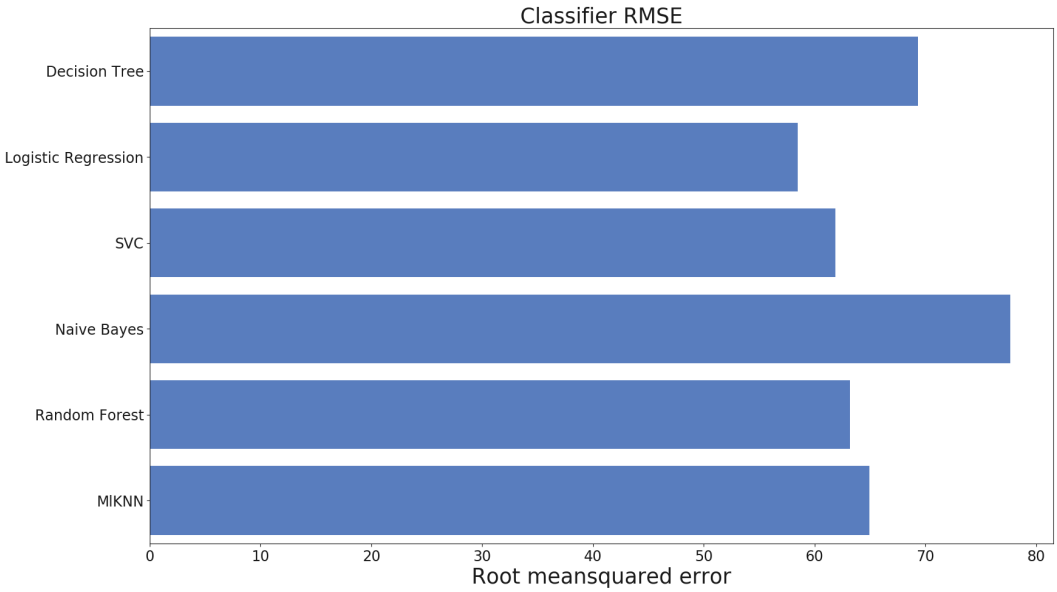


Figure 5.8: Classifier RMSE for SVC(kernel=linear), Regression, MlKNN, RandomForest, DecisionTree, GaussianNB paired with the proposed co-training classifier

the earlier chapter, multiple-label co-training resulted in better results, but it was unknown why the confident samples were selected for training. In the experiment, we demonstrate that, by combining both algorithms - similarity and continuous learning - we are able to determine why certain stories are assigned certain labels. Furthermore, the SVD method is used to confirm if the newly assigned labels/set of labels actually belong to the story because there is content that matches in the stories with similar labels. As a result, we are able to classify and label stories without any additional human involvement. In this way, one can discover patterns and collect information that is similar, which will assist in the labeling of the huge amounts of unlabeled information being collected and stored on the internet. Therefore, experts in a domain would be able to reduce the amount of time and effort they would devote to manually labeling each story. This would also help to eliminate the problem of wrongly categorizing victim stories, which can severely impact the rehabilitation process. Currently, the algorithm is evaluated in the human rights domain, but collaborative filtering is domain-independent. Thus, it is believed that this approach has the potential to be tried

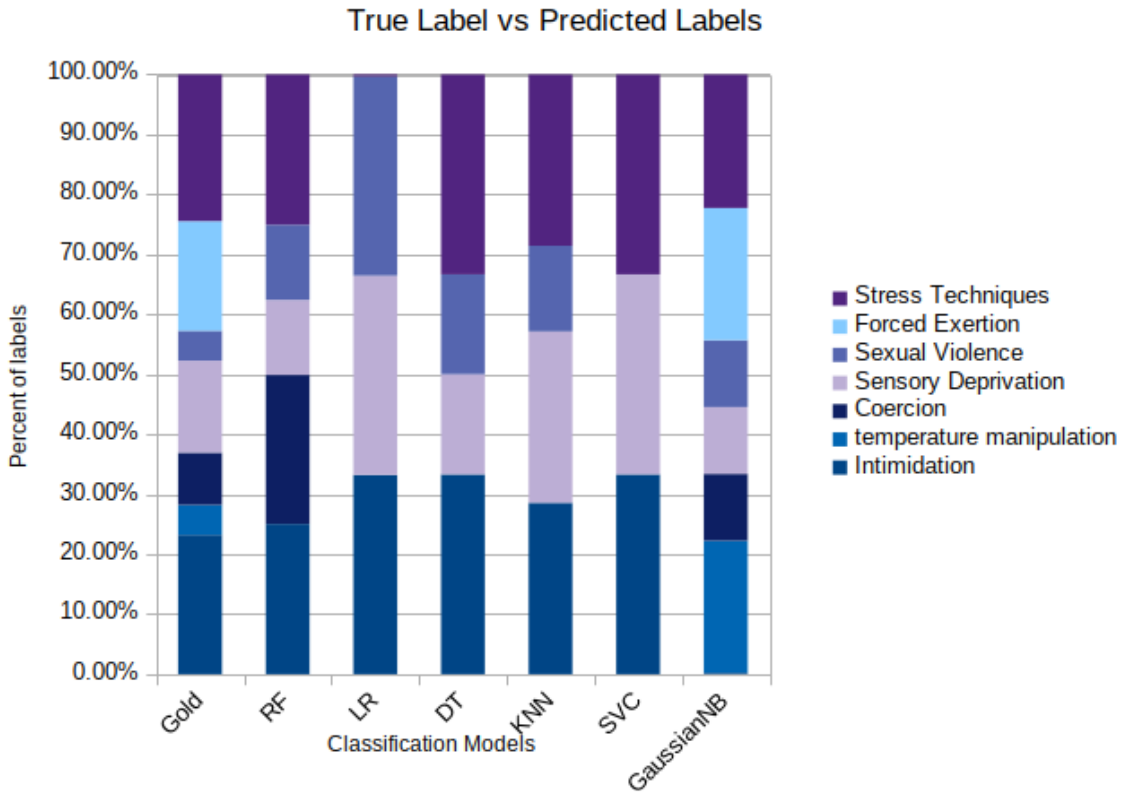


Figure 5.9: True values(Ground truth) comparison with predicted values

also be used in several other domains.

In this method, the cold start problem may be overlooked. This is because all ranking and factorization would only work if historical data existed on which to base the relationship matrix. The algorithm, however, would have difficulty generating these matrices without such information present. In the next chapter, we will look at stronger similarity measures that can provide insight into how patterns form between labels and existing stories.

Chapter 6

Recommendations for enhancing co-training

The previous chapter demonstrated how we can use matrix factorization to rank predictions made by two classifiers as part of the co-training algorithm. In addition, a recommender system was used to determine the score of each prediction and to select the label or set of labels that predicted the most accurately. To make this method effective, there needs to be enough data for the classifiers to determine a user-item relationship or, in the case of this thesis' experimental dataset of survivor stories, a label-stories relationship. However, gathering enough data is usually time-consuming and very challenging. Ratings and feedback are the foundations of the recommender system. Consequently, improving the recommendation accuracy for items that are new or rarely rated as well as for inactive or new users presents its own challenges. It is impossible to obtain user feedback at all times. Therefore, as a response, this chapter explores ways to extend the co-training matrix factorization framework by using recommendations that provide feedback or ratings at any time as seen being hypothesized in Kihlman and Fasli (2019). Thus, we avoid the cold-start problem of low or no ratings and provide recommendations. By integrating several similarity-based algorithms, for instance Euclidean distance, correlation, cosine angle, and Manhattan distance, as well

as content filtering, the evaluation works even better and the most efficient algorithm will be selected. This chapter describes a co-training system that combines matrix factorization with similarity measures to boost prediction confidence and accuracy.

It has been shown in Zhu et al. (2007) that using the single view co-training algorithm with matrix factorization allows both classifiers to perform similarly, as they were trained on a large set of labelled samples. Furthermore, a growing number of large and small scale online companies, Fleder and Hosanagar (2007), are implementing recommendation systems to increase user interaction and enrich the appeal of their user content based on user preferences. A recommender system has the potential to promote customer loyalty since it learns more about the users, thereby encouraging them to purchase additional items, increasing company profits, and encouraging them to subscribe or advertise in order to ensure long-term use Schafer et al. (2001). Aiming to enhance co-training, this chapter harmonises similarity measurements with the method using large unlabeled data in order to improve recommendation quality. Typically, recommender systems provide recommendations based on a ranking or some feedback from the users, and there are times when this feedback is not sufficient for the recommendation. In such cases, the recommender algorithm may be better suited to co-train based on matrix factorization on sparsely labelled feedback data, along with a large unlabelled dataset to form an adequate set for decision making. After the new rankings are identified, they are added to the labelled data. The training set is then trained using this updated labelled data and predictions can be generated to classify the remaining un-annotated data. These coming sections describe in detail how the algorithm is implemented and the similarity measures used, while the Experiment1 section evaluates the algorithm on a generated dataset while the Experiment2 section evaluates real-world scenarios and discusses the performance of the proposed algorithm. The chapter closes with a critique of the framework as well as suggestions for further improvement.

6.1 Method

This chapter seeks to address the problem of not having enough data to formulate recommendations. A variety of similarity measures are used to compare the stories with similar labels in the minimally labelled dataset and then labels for the stories that have the best confidence are chosen. It can then create a stories/label relationship matrix for training two classifiers with this information. As such, we utilize continuous learning methodology for labeling unlabeled data that requires limited manual intervention. Therefore we extend the algorithm described in the previous chapter by using similarity methods to demonstrate the ability of co-training to recommend similar labelled stories and classify the new story based on the labels with the highest similarity scores. We can address this problem by making changes to the co-training process as below:

- Two base classifiers are trained on a single view of the data to provide the initial set of labels. Then using Matrix Factorization, labels are ranked
- Use various recommendation algorithm to provide similar stories to compare the label rankings to find the appropriate label/labels for the story
- Calculate the similarity score for the label based on the similarity model. Select the label that has the highest score.

6.1.1 Ranking using the single view

Co-training assumes that the dataset X consists of two views $X = X_1 \times X_2$ with their respective feature partitions X_1,X_2. The two views must satisfy two conditions:

- Both have to be sufficient within the given hypothesis class H i.e. there exist two hypothesis h_1,h_2 H having low error on X_1,X_2 respectively.
- They need to be class-conditionally independent, i.e. for a given $x = (x_1, x_2) \in X_1 \times X_2$ with label $y \in Y$, $p(x_1|y)p(x_2|y) = p(x_1, x_2|y)$.

A multi-label co-training classification model was implemented in the previous chapter. Therefore, it is necessary to take into account a more relaxed interpretation that attempts to maintain a balanced distribution of positive and negative samples across the labels. In the notation presented in Sechidis et al. (2011), let's consider a data set, D, annotated with a set of labels, $L = \lambda 1, ..., \lambda q$, a desired number k of disjoints subsets $S_1, ..., S_k$ of D, and the desired proportion of samples $r_1, ..., r_k$ in each of these subsets. The desired number of samples at each subset S_j is denoted as c_j and is equal to $r_j \cdot |D|$. The subsets of D and S_j that contain positive samples of label λ_i are denoted D_i and $S_i j$ respectively. Even when there are multiple labels, they remain mutually exclusive. By using the conditional independence between the labels, we are able to divide the whole dataset into two feature views X_1 and X_2 on which we can train two independent classifiers. Thus conforming to Blum's assumptions about co-training.

As part of the deployment of classifiers, we use state-of-the-art classifiers like linear Support Vector Machine (SVM), linear Regression, and Naive Bayes trained by Stochastic Gradient Descent (SGD). As reviewed by Sivakumar and Rajalakshmi (2019) a learning algorithm for instance SGD estimates the gradient of the loss function each time a sample is taken and the model is updated along the way with a decreasing learning rate. It is regularized by adding a penalty to the loss function. In our case, all the labels are categorical, meaning they belong to some category and are in text format. An integer encoding is not adequate for categorical variables that lack such a relationship. Our text labels are represented by OneHotEncoding Seger (2018), which generates a binary representation. In other words, categorical data must be transformed into numerical data.

With the aid of human rights experts, the violation a survivor experienced can be broadly classified into the following categories in the survivor stories dataset: -

- Sensory deprivation
- Stress techniques

- temperature manipulation
- Coercion
- Intimidation
- Sexual violence
- Forced Exertion

As described in chapter 3, we have seen each story can be grouped into one or more of these labels. So now we need to convert these categories of text labels into integers via the OneHotEncoding method. The labels after OneHotEncoding are as seen in figure 6.1. Each label category if it exists for that story in the label set is represented as 1 and the rest as 0. Thus, the story with id 1 has labels now represented as 1,1,0,0,0,1,1 which translates to Sensory deprivation, Stress techniques, Sexual violence, Forced Exertion.

Initial ranks have been associated with each label for every story present in the labelled set. Deploying the extended version of the co-training algorithm explained in chapter 4, the two SGD classifiers ($x1$, $x2$) are trained using this labelled set (L). A random sample from the unlabelled data set (U) were taken (U') to train the classifier and predict labels for U'. These labels are then ranked using the Matrix Factorization method.

6.1.2 Similarity methods

Recommender systems as early as 1997 as demonstrated by Resnick and Varian (1997), were defined as ones in which people provide recommendations as inputs, which the system then aggregates and directs to appropriate recipients. The term now has a broader connotation, describing any system that produces individualized recommendations as output or has the effect of guiding the user in a personalized way to interesting or useful objects in a large space of possible options. Such systems have an obvious appeal in an environment where

storyid	story	Sensory Deprivation	Stress Techniques	temperature manipulation	Coercion	Intimidation	Sexual violence	Forced Exertion
1	Hooding with sandbags/cement bags	1	1	0	0	0	1	1
2	Blackened goggles	0	1	1	0	0	1	
3	Forced silence/ duct tape on mouth	0	1	1	1	1	0	0
4	Prolonged sitting in required position	1	0	0	0	1	1	0
5	Prolonged standing/wall standing	1	1	0	1	1	1	1
6	Sexual violence to genitals	1	0	0	0	1	1	1
7	Molestation	1	1	1	0	0	1	1
8	Penetration using instruments	0	0	1	1	0	1	0
9	Pressed on hot surfaces	1	0	1	0	0	0	0
10	Stamping on victim	0	1	0	1	0	0	1
11	Dragging victim along ground	0	1	1	1	1	0	1
12	Simulated drowning	0	0	0	0	1	0	1
13	Verbal abuse	0	0	1	0	1	1	1
14	Detention in unbearably hot locations	1	1	1	0	0	0	0

Figure 6.1: Categorical labels converted to binary representation using OneHotEncoding

the amount of online information vastly outstrips any individual's capability to survey it. Today, recommender systems are integrated into e-commerce sites for instance Amazon.com and Netflix Zhang et al. (2021). Recommender systems differ from traditional information retrieval or search engines as they focus more on individualization and interest and value. Search engines have semantics related to "matching." That is, they should return all the items that match the query and rank them based on their degree of match. Search engines use such techniques as relevance feedback to refine their representation of the query of the user, providing a simple form of recommendation. The need to combine recommendation techniques to achieve peak performance is common to recommender systems research. Of the three recommendation systems, collaborative filtering is the most popular, successful, and widely used Sahu and Dwivedi (2019); Zhang et al. (2016); Laishram et al. (2018). Content-based filtering and hybrid filtering are the other two. This method is simple, efficient, and works well because it produces near accurate results. However as said earlier, it still required historical data to improve the efficiency of the recommendations. Several similarity improvements have been proposed that potentially look at improving recommendation accuracy. Among them are Ramezanis model Ramezani et al. (2021), the multi-level collaborative filtering similarity Polatidis and Georgiadis (2016), and the Triangle Multiplying Jaccard (TMJ) similarity Sun et al. (2017). By analyzing users' behavior, these methods enhance the accuracy of recommendations. In a similarity based recommender system, the purchase histories of users who purchase or like the same items were analyzed, and similarities between the users were established. Based on this correlation, recommendations were made, assuming that two users with a similar purchase pattern would like to see future recommendations made by their peers.

If we apply the same approach to the dataset used for this thesis, we will see that the recommender ascertains whether there are any other similar labelled stories present in the already labelled sample for the story with the newly predicted labels. Similarity between the stories determine the closeness of the label with their semantic context. According to

Bolognesi and Aina (2019), closeness between any content can be established by observing the order of the words or phrases without understanding the meaning behind them. Balogh et al. (2020) states that semantic content is understanding the meaning of the words and whether the two samples having different words mean the same. The easiest way to find similarities between textual data is by creating a feature vector and use distance measures to determine closeness between these features. Although there are several state of the art recommendation algorithms present as presented in their review by Vijaymeena and Kavitha (2016), this chapter concentrates on the following models as according to the authors these models are better tailored to the nature of data being classified and compared by this thesis.

- Cosine distance-based - Cosine distance function is commonly used to find similarities between different documents. Cosine similarity measures the degree of angle between two documents, in a multidimensional space. There is a higher chance of similarities between documents when the angles are smaller. As seen in equation 6.1 cosine similarity is seen as the dot product of the two vectors divided by the product of the two vectors' lengths (or magnitudes).

$$cosSim(A,B) = \frac{\sum_{i=1}^{n} AiBi}{\sqrt{\sum_{i=1}^{n} Ai^2}\sqrt{\sum_{i=1}^{n} Bi^2}} \tag{6.1}$$

  where

  - $cosSim(A,B)$ is the cosine similarity score between 2 users A, B
  - Values range between -1 and 1, where -1 is perfectly dissimilar and 1 is perfectly similar

- Euclidean distance-based - Euclidean distance function is just the distance between two points. But it also signifies that similar items will lie in close proximity to each

other if plotted in a multi-dimensional space.

$$d(A,B) = \sqrt{\sum_{k=1}^{n} (A_k - B_k)^2} \tag{6.2}$$

where

- $d(A,B)$ is the Euclidean distance between 2 users A, B

Equation 6.2 defines the Euclidean distance between two points in one, two, three, or higher-dimensional space where n is the number of dimensions and A_k and B_k are components of A and B respectively. To convert this distance metric into the similarity metric, we can divide the distances of objects by the maximum distance, and then subtract it by 1 to score the similarity between 0 and 1.

- Manhattan distance-based - The Manhattan distance function computes the distance from one data point to the other if a grid-like path is followed. The Manhattan distance between two items is the sum of the differences of their corresponding components. This helps in determining how similar the documents are to each other if placed on a grid. The distance function in an n-dimensional space can be calculated as shown in equation 6.3.

$$d(A,B) = \sqrt{\sum_{k=1}^{n} (A_k - B_k)^2} \tag{6.3}$$

where

- $d(A,B)$ is the Manhattan distance between 2 users A, B

Since this is a distance metric, it will have to be converted to a similarity metric as so:

$$sim(A,B) = \frac{1}{1 + d(A,B)} \tag{6.4}$$

where

- $sim(A,B)$ is the Manhattan similarity score between 2 users A, B

  - $d(A,B)$ is the Manhattan distance from equation 6.4

- Jaccard similarity - Jaccard similarity is the ratio of the number of common phrases to the number of unique words in both documents as seen in equation 6.5

$$sim(A,B) = \frac{\|A \cap B\|}{\|A\| + \|B\| - \|A \cap B\|} \tag{6.5}$$

  where

  - $sim(A,B)$ is the Jaccard similarity score between 2 users A, B
  - A represents the set of all items preferred by user A
  - B represents the set of all items preferred by user B

- Pearson's similarity - This measure tells us how much two items are correlated. The higher the correlation, the higher the similarity. Pearson's correlation coefficient is calculated as :

$$p(A,B) = \frac{\sum_1^n A_i B_i - n\overline{AB}}{(n-1)\, s_A s_B} \tag{6.6}$$

  where

  - $p(A,B)$ is the Pearson's coefficient correlation score between 2 users A, B
  - s_A and s_B is the standard deviation of the samples

A similarity score indicates how closely two samples are related. The dissimilarity measure, on the other hand, indicates how distinct the data objects are. The similarity measure is usually expressed numerically: It increases when the data samples are more similar. Usually, it is expressed as a number between zero and one by conversion: zero means low similarity (the data objects are dissimilar). 1 means very high similarity. Euclidean, Manhattan and Cosine Distance Measures can be used for calculating document dissimilarity. Since similarity is the inverse of a dissimilarity measure, they can also be used to calculate document similarity.

6.1.3 Label selection

Based on the newly predicted story labels, each similarity model is run independently to determine the most similar content in all documents. We gather and rank the common phrases and words in documents by comparing the matrix factorization scores. Based on the similarity score for each label, a weighted average is calculated based on the n-most similar documents. Then all labels with a score higher than 0.5 are added to the current document's labels. Those are added as positive labels to the existing set of labels.

$$lscore_i = \frac{\sum_{c=0}^{count} sim(story, story_c) * lscore_{ci}}{count} \tag{6.7}$$

where

- $lscore_i$ is the score for the i_{th} label for the document;
- $count$ is the number of documents we know the labels for;
- $story_c$ is the existing document that is being compared with;
- $sim()$ is a function that returns a similarity between two documents;
- $lscore_{ci}$ is the i_{th} label for $story_c$.

6.2 Similarity based extension to the algorithm

Figure 4 shows the revised co-training algorithm after incorporating the additional steps mentioned in the earlier section. Comparing the earlier and revised methods, the early method relied on recommendations to find more stories associated with existing or relevant datasets. Identifying the right kind of rehabilitation strategy for a victim could be done by comparing similar stories from earlier samples. According to the revised method, recommendations are used to generate dummy data samples to be inserted into the limited

Given:
a set L of labelled training samples with only some samples ranked by experts
Use MF rank the remainder of the training samples
Create a pool U' by choosing u at random from U
Divide training set X into x1 and x2 randomly
for $i \leftarrow 0$ to k by 1 do
    Use MF rank U' samples
    Use L to train h1 on the x1 portion of x
    Use L to train h2 on the x2 portion of x
    Calculate similarity similarity score
    Select top 10 similar stories for U' based on the score
    Compare similarity score with prediction score for U' by h1 for x1
    Select the highest score in x1 as positive samples
    Compare similarity score with prediction score for U' by h2 for x2
    Select the highest score in x2 as positive samples
    Label p positive labels from x1
    Label p positive labels from x2
    Add these labelled samples to L
    Randomly choose 2p from U to replenish U'
    Continue until U is exhausted
end

Algorithm 4: Revised Co-training Algorithm

labeled set for the purpose of building a user/item relationship matrix and predicting accurately. Consequently, the cold start problem will no longer be a problem even when there is insufficient historical data.

Table in 6.1 shows the initial set of labels annotated with the assistance of human rights experts. A total of 14 stories are shown with the appropriate labels. Human rights experts have annotated about seven labels for each story as shown in the table. The SGD classifiers would be trained on this labeled set, to predict the labels for all the unlabeled stories. Based on the original labels and the various similarity methods used in the co-training algorithm, an additional 20 new stories were added with appropriate labels as illustrated in figure 6.2. By labeling the stories, we demonstrate how similarity measures help discover similar stories, thereby assisting the co-training algorithm to learn and predict more accurately at each iteration.

story	Sensory Deprivation	Stress Techniques	temperature manipulation	Coercion	Intimidation	Sexual violence	Forced Exertion
Hooding with sandbags or cement bags	1	1	0	0	0	0	1
Blackened goggles	1	1	0	0	0	0	1
Forced silence or duct tape on mouth	1	1	0	0	1	0	0
Prolonged sitting in required position	1	1	0	0	0	0	0
Prolonged standing or wall standing	1	1	0	0	0	0	0
Sexual violence to genitals	0	0	0	1	1	1	0
Molestation	0	1	0	1	1	1	0
Penetration using instruments	0	0	0	1	1	1	0
Pressed on hot surfaces	1	0	1	0	0	0	1
Stamping on victim	1	0	0	0	1	0	0
Dragging victim along ground	1	0	0	0	1	0	0
Simulated drowning	1	0	1	0	1	0	0
Verbal abuse	0	1	0	0	1	0	0
Detention in unbearably hot locations	1	0	1	0	0	0	1
Amal	1	1	0	0	0	0	1
Adam	1	1	0	1	0	1	0
Hassan	1	1	0	0	1	0	0
Azra	1	1	0	0	0	1	0
Akram	1	0	1	0	0	1	1
Ali	1	1	0	0	1	1	0
Haben	1	1	1	0	0	1	1
Gebriale	1	1	0	0	1	0	0
Fajer	1	1	0	0	0	1	0
David	1	0	1	0	0	1	1
Jons_story	1	1	0	0	1	0	0
Jorge	1	1	0	0	1	0	0
Esme	1	1	1	0	0	0	1
Bushrah	1	1	1	0	0	0	1
Jean	1	1	1	0	0	0	1
Feven	1	1	1	0	0	0	1
Janas	1	1	0	0	1	0	0
Imad	1	1	0	0	1	1	0
Jana	1	1	1	0	0	0	1

Figure 6.2: Additional stories with new labels from the unlabelled set added

6.3 Experiments

For the purpose of evaluating this method, two separate experiments were conducted. As part of the first experiment, we deploy the method on the survivor stories dataset to emphasize the focus of the thesis and use a minimally labeled training set to predict new labels. In the second experiment, an actual dataset is used to demonstrate that the method is domain-independent and can be adapted to any domain that has a small labeled training set.

6.3.1 Experiment 1 - Survivor stories dataset

There are two parts to executing the algorithm. Co-training and recommendations are used to create a comprehensive labeled set from a small unlabelled subset of the test data set in the first phase. The training set is then trained with the newly defined labels. The second phase of the experiment involves running the algorithm on the remainder of the unlabeled set to predict labels and test the accuracy of the predictions. To determine which model is most efficient, the algorithm is run for all similarity methods.

The unlabeled set is a 10 MB collection of 400 testimonies and survivor stories scraped from the Internet and stories of victims of torture from Business and Centre (n.d.). This data was pre-processed ad normalised to have meaningful content for classification. The normalization of this data includes removal of all the HTML/css/link tags, removal of stop words and punctuation and all text is converted into lower case.

Since this chapter addresses the problem of multiple labels, popular classifiers like LogisticRegression, LinearSVC, DecisionTrees, and Random Forest along with the extended co-training classifier that we demonstrated in chapters 3, and 4 are used to compare the predictions given during the training phase. This means that all the labels predicted for each story by the algorithm, are compared to the labels predicted by the popular classifiers. This evaluation is repeated for all the similarity models and their results are compared.

6.3.2 Metrics

To determine the effectiveness of a multi-label classifier, the prediction of the most relevant documents for each category to quantify the quality of the predicted values has to be compared. The following measures have been used for the classifiers' evaluation:

- earning curves are used to observe the bias-variance characteristics displayed by the different learning models;
- hamming loss, as it calculates the number of incorrect labels in relation to the total number of labels, it offers an effective measure for multi-label classification
- T Root Mean Square (RMSE) is a standard metric used for evaluating the accuracy of predicted ratings against actual ratings;
- McNemar test checks if two classifiers disagree in the same way.

6.3.3 Results

The results would be evaluated using the following evaluation methods -

- **Evaluation based on Hamming loss** - Hamming loss determines the number of times the labels get incorrectly classified. The objective of the proposed algorithm is to minimize the loss occurred during the learning phase. The popular classifiers are trained using 100% of the labelled set while the Matrix Factorisation co-training and the similarity-based co-training were trained on 10% of the labelled set. Hamming loss of the popular classifiers was considered as the baseline to compare the performance of the rest of the classification techniques. As seen from figure 6.3, similarity-based co-training method using Pearson's coefficient has been consistently displaying the least number of labels getting mis-classified throughout all the base classifiers.The Manhattan distance-based similarity method comes in as a close second best method using Hamming loss as the evaluation metric.

- **Evaluation based on RMSE and ROC** - RMSE is a way to effectively cross-validate the performance of the classifier to check the outcome of the true positives and false positives of the predictions. As seen from table 6.1, Decision Tree although displays the highest accuracy, it has a very low recall and precision value. This indicates that the number of false positives and false negatives is quite high which is also aptly indicated by its ROC curve. The area under the curve lied steady around the 0.5 mark which establishes that predictions could be just random. The overall accuracy of the similarity-based classifiers is hovering almost around the same mean along with being complemented by the respective recall and precision. For example, classification using Pearson's coefficient as the similarity method, has a consistent high accuracy, recall, and precision and is aptly visualized in its ROC curve.

- **Evaluation using the McNemar test** - McNemar test checks if two classifiers disagree in the same way. It uses a contingency table to determine whether the count of correct/incorrect labels predicted by the classifiers are the same or different. The test was carried out two times. One compared popular classifiers with co-training using similarity methods shown in table 6.3 and the other compared co-training with

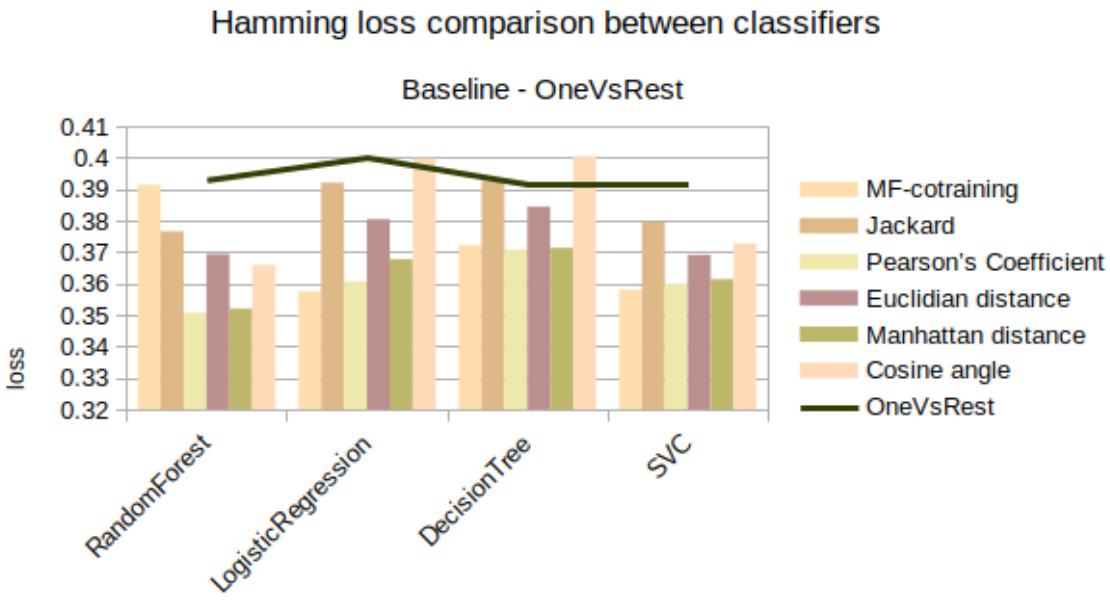


Figure 6.3: Comparing classifier performance on hamming loss

co-training using similarity measures shown in table 6.2. The default assumption, or null hypothesis, of the test, is that the two cases disagree to the same amount. If the null hypothesis is rejected, it suggests that there is evidence to suggest that the cases disagree in different ways, that the disagreements are skewed. As can be seen from both tables, the test confirms that apart from the Pearson's similarity methods, the rest have very little difference between the predicted values by both classifiers. Thus rejecting the null hypothesis.McNemar test checks if two classifiers dis- agree in the same way.

Overall it is observed that the similarity-based recommending co-training algorithm displays enhanced performance as compared to its peer state of the art classification methods. Amongst the similarity-based models, the one outperforming the rest in all aspects is the Pearson's coefficient model.

Classifier Name	RMSE	Precision	Recall
DecisionTree	0.679	0.493	0.376
SVC	0.6408	0.560	0.469
RandomForest_euclidean	0.618	0.589	0.537
LogisticRegression_euclidean	0.610	0.605	0.525
DecisionTree_ct	0.671	0.501	0.633
RandomForest_ct	0.660	0.516	0.538
LogisticRegression_ct	0.598	0.61	0.578
SVC_ct	0.598	0.61	0.578
LogisticRegression_euclidean	0.616	0.585	0.553
DecisionTree_euclidean	0.611	0.592	0.553
SVC_euclidean	0.607	0.598	0.566
RandomForest_euclidean	0.602	0.605	0.566
DecisionTree_manhattan	0.619	0.610	0.447
LogisticRegression_manhattan	0.618	0.616	0.428
SVC_manhattan	0.6021	0.663	0.415
RandomForest_manhattan	0.596	0.671	0.431
LogisticRegression_pearsons	0.616	0.592	0.538
RandomForest_pearsons	0.6053	0.605	0.568
SVC_pearsons	0.605	0.605	0.568
DecisionTree_pearsons	0.598	0.620	0.554
DecisionTree_jaccard	0.636	0.563	0.501
LogisticRegression_jaccard	0.6262	0.576	0.528
SVC_jaccard	0.6177	0.587	0.551
RandomForest_jaccard	0.6163	0.592	0.539

Table 6.1: Classifier comparison based on RMSE,Precision and Recall

similarity methods	LogisticRegression	DecisionTree	SVC	RandomForest
Jackard	0.000194	0.0536	0.274	5.83e-31
Manhattan Distance	0.330	0.00317	0.754	1.39e-30
Cosine Angle	4.72e-06	0.000140	0.327	7.28e-23
Pearson's	0.545	0.266	0.765	2.70e-36
Euclidian Distance	1.16e-05	0.000412	0.00393	1.55e-23

Table 6.2: Results for McNemar's test on cotraining vs. similarity

6.3.4 Experiment2 - Bonusway

Bonusway, is a Finnish online service that pays a bonus for all online purchases made by its users. To extend their vision of partnership marketing, Bonusway intends to gather knowl-

similarity methods	LogisticRegression	DecisionTree	SVC	RandomForest
Jackard	0.0308	0.0052	6.0086e-06	4.69e-07
Manhattan Distance	0.187	0.00085	1.003e-06	0.00047
Cosine Angle	5.5568e-11	.6026e-10	0.2831	0.02087
Pearson's	0.744	0.305	4.21e-07	6.01e-08
Euclidian Distance	0.000160	0.0014	0.0205	0.037

Table 6.3: Results for McNemar's test on popular vs. similarity

edge about its users and then be able to rank the increasing amount of offers for each user, to optimize what offers they show each user in their online feed. Bonusway is one of the leading e-commerce cash-back services in Europe and has about 33 countries participating. There are offers and bonuses offered by online stores in each country. If a user visits their website through Bonusway, the online store recognizes that user as a Bonusway user and pays them a commission on purchases. Stores are also known as campaigns. There are two ways Bonusway generates revenue. One for every click the user makes on a particular campaign, regardless of whether or not they are offered anything related to it. The second way is through the actual purchase made by the user for a campaign that either has or does not have an offer associated with it. These offers are reduced prices or other promotions the retailer is offering at that time.

They implemented an in-house recommender system to improve sales. However, they have faced some challenges with that system and those have been described below:

- data scarcity issues where there will be no data directly connected to any offers for the majority of the users.
- A good amount of the users have not made any purchases or clicked on any link, a typical cold start issue.
- New offers are created, but no relationship can be established between user purchases and offers.

The objective of the algorithm proposed in this chapter was to overcome challenges

similar to those faced by Bonusway, and hence the developed approach was deployed in their production environment to evaluate the effectiveness of the personalised recommendations.

6.3.5 Dataset

The dataset used for the recommender system is a collection of the following based on each country -

- all active stores that generate offers
- user-specific click events for eg, click on links in the email, store links, offer links
- purchases made for an offer on a store
- user data, specifically gender, birth year, and country of residence

The tests were run on a total of 4.2 million unique users, spread over 11 countries all over Europe. The smallest country Estonia has 13 thousand users and the largest, Russia has 2.4 million users. The test to control ratio was 9:1. The tests were split over 14 email events spaced roughly one week apart. The test and control groups were also chosen to have an equal ratio of gender, age, and activity (clicks, purchases). The control group were sent randomly chosen offers.

6.3.6 Performance evaluation

The evaluation of the algorithm is based on the effectiveness of the recommendations to display an optimised set of offers for each user in their feed. As well as demonstrating how the recommender has improved its understanding of the user base.

The amount of revenue generated by Bonusway after deploying the algorithm is based on the following metrics:

- **Revenue per user clicks** - A conversion is when a user performs a desired action (being tracked) on the Bonusway website. This could be buying a product, downloading

a file, clicking a link, or any other desired action. Typically conversions are linked either directly or indirectly to revenue. This metric is a good indicator of campaign performance, especially if when there are multiple campaigns promoting the same products, where a comparison can be made between each campaign. Every customer at Bonusway receives personalized recommendations based on their purchases. User movement is tracked to generate revenue as well as to collect insights to feed the algorithm with new data to improve predictions.

- **Revenue per actual purchase** - A conversion occurs when the user purchases an item recommended in the e-mail; this is the revenue generated. In addition to increasing sales, this metric signals user preferences to send similar recommendations. These recommendations may also be sent to users who show similar purchasing patterns.

- **Re-engage inactive users** - Maintain the correct recommendations by displaying incentives, contests, discounts, and coupons in Bonusway's email messages to encourage users to open them. It would then result in an increase in click-through rates and/or revenue per purchase.

6.3.7 Results

The algorithm has been running in the production environment since February 2019. Every week the recommender collects data for the previous week and starts learning iterative using the new feed. A fresh set of user/store recommendations are generated and mapped against the new offers for each store. To describe the performance of the recommender system, we used December 2021 data and results.All metrics showed an overall improvement in performance, but with a large variation between countries. The most consistent metric was the activation rate, with an average improvement of 2.2% and a standard deviation of 5.2 percentage points. The least consistent being the revenue, with an average improvement of 6.12% and a standard deviation of 15.85 percent points. The click-rate, while having a

larger variation than the activation metric, also showed a larger improvement and it is the only metric that only has one country below 0% increase (at 3.5% decrease), while the rest have between 3.3% and 31.7% increase in click rate. This gives an average of 12.21% and a standard deviation of 10.3 percentage points. This is summarised in table 6.4.

Metric	Mean	Std Dev
Clicks	12.21%	10.3%
Activation	2.2%	5.2%
Revenue	6.18%	15.85%

Table 6.4: The mean and standard deviation for improvement for each metric over the 11 countries

Looking at the actual numbers (normalised by user base per country), and calculating the p-value using Z-scores, we see that none of the metrics are statistically significant. This means that we do not know whether the improvement seen is due to random chance, or due to the algorithm being able to make better recommendations. These results are summarised in table 6.5.

Metric	Test Mean	Std Dev	Control Mean	p-value
Clicks	0.0219	0.0167	0.0199	0.345
Activation	0.0110	0.0101	0.0108	0.471
Revenue	0.0521	0.0340	0.050	0.406

Table 6.5: The mean and standard deviation for each metric over the 11 countries, normalised by users per country

There is no indication in the data that the size of user-base or amount of valid offers available affects the results one way or the other. The breakdown of countries with a weighted average of valid offers, clicks, activation, and revenue is plotted in figure 6.4.

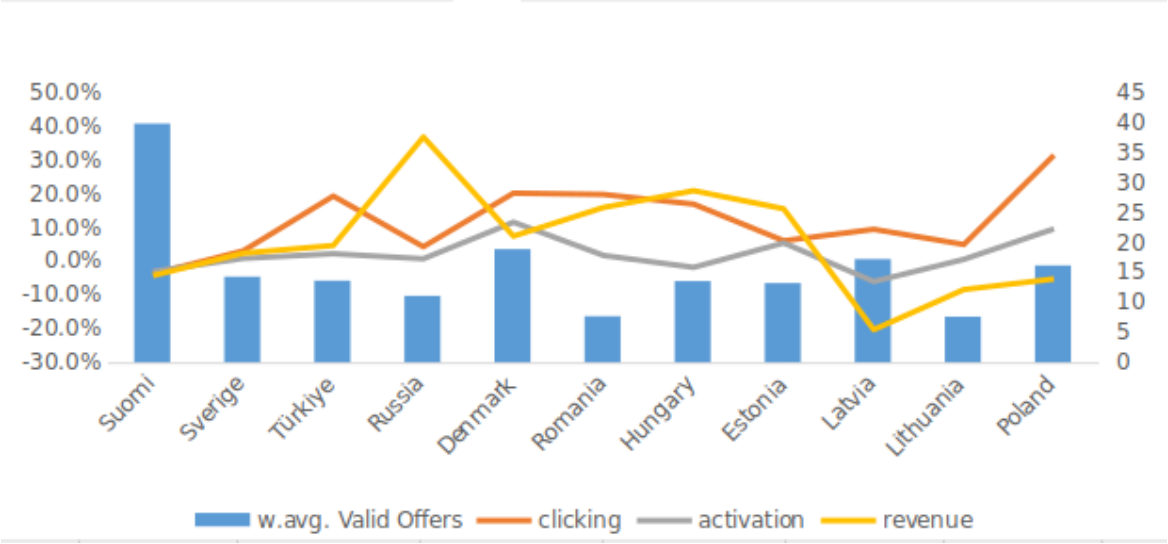


Figure 6.4: Effectiveness of the algorithm based on clicks, revenue and purchase for all countries

6.4 Conclusions

As demonstrated in chapter 4 the revised co-training algorithm made use of matrix factorization to rank the labels and predict labels for the unlabeled dataset. In the case of the survival stories dataset, it made use of identifying underlying relationships between labels and stories. This relationship occurs when some documents are associated with certain labels based on historical data. It demonstrated good results but struggled when there was no or insufficient historical data. Therefore in this chapter, we investigated the addition of similarity-based recommendations to the co-training algorithm to mitigate the cold start issue of insufficient data. Here relationships were determined when stories are similar in nature, either by content or semantics.

In the case of the victim's dataset (Experiment1), a model that optimizes recommendation performance based on similarity measures under cold-start conditions when only a few annotations and their ranking are available. When compared with the traditional generic similarity measure, the suggested measure analyzes the rankings data in the context of label recommendation, and thus, uses the ranking data in cold-start conditions to predict new labels more effectively. For completeness, experiments are conducted with multiple datasets

(survivour stories and Bonusway).

The primary contribution of the presented model is a novel similarity measure for collaborative filtering (CF) methods that can replace traditional similarity and distance measures, for instance Pearson's correlation and cosine. In addition, most studies that address the cold-start problem present hybrid approaches that combine content-based information with ranking data, focusing on the cold-start situations where no ratings are available at all. The framework replaces existing CF methods with similarity measures that leverage the small labeled set to create new ranks and labels and update the labeled set with them, as opposed to improving cold-start situations by using semi-supervised co-training that leverages the small labeled set by creating new ranks and labels and updating the labeled set with them. Thus, allowing classifiers to be trained on and predict with more accuracy using an updated labeled set. In addition to its main contribution, this model assigns one or more labels to the story to give a comprehensive view of the abuse suffered by a victim or, as with Bonusway, the reach of a product for a buyer to appreciate.

Based on the results, we conclude that similarity is the most useful measure for suggesting stories based on the labels. The similarity is particularly useful when there is insufficient information to create the story/label relationship matrix for the survivor stories database used for evaluating the model. Co-training increased the likelihood that a new story would be labelled correctly by using the best similarity. As a result, it was found that using this method mitigated the problem of having less labelled samples to generate the relationship matrix. Moreover, when coupled with similarity-based recommendations, co-training improved performance on experimental datasets, as the probability of confident samples predicted by this method demonstrated higher precision compared to standard classifiers. Compared to its peers, Pearson's coefficient of similarity-based co-training produced the best results. A further strength of the algorithm is its capability to work even when there is limited data availability and yet train two classifiers and extend the labelled dataset.

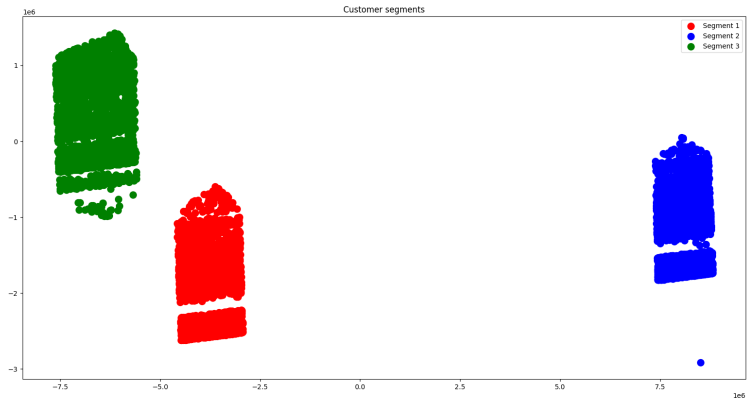
After applying the algorithm to the Bonusway data, several interesting relationships

were revealed, including:

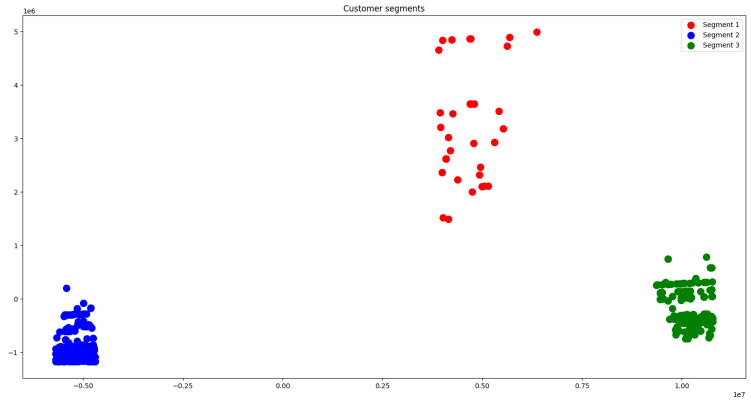
- **The similarity in Behavior Data**- Users behavior data is useful information about the engagement of the user on the product. It can be collected from clicks, and purchase history. Three distinct customer segments are represented in the figures 6.7 and 6.6 based on buying behavior. We can interpret buying behavior by being direct and making the purchase or by clicking other options and deciding to make the purchase later. For example, Ukraine shows fewer clicks, but a much higher number of purchases, while Sweden and Denmark show a consistent pattern of first surveying the market, then making a purchase. The other noteworthy thing is that the customer segments do not overlap. As a result, opportunities for growth and expansion are present since retailers do not dominate customers. Therefore, improving sales can be achieved through appropriate market positioning, pricing strategies, and coordinated sales & marketing efforts, as well as promotions and bundles.

- **Similarity in Demographic Data** - User demographic information is related to the users personal information for instance age, education, income, and location. Figure 6.7 shows the top 20 customers that have generated store-wise revenue. The customers have been categorized by age and gender. This figure shows that, between 2019 and 2020, Denmark's highest revenue is generated by store 43608 by a male born in 1984. While 47514 seems to be a favourite for both female and male customers who want to make purchases during this time period, regardless of their age. Potentially, both of these stores could be recommended to other customers to increase sales. Swedish men appear to prefer store 17665, while Swedish women favor store 29295. A sales strategy might include recommending these stores or products similar to those found at these stores to others. It appears that Ukraine shows a relatively quiet trend, where only two stores dominate irrespective of gender or age.

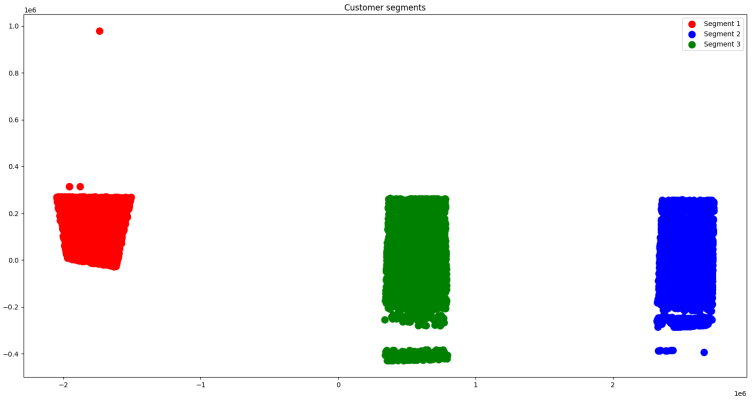
- **Similarity in Product Attribute Data** - Product attribute data is information related



(a) Sweden

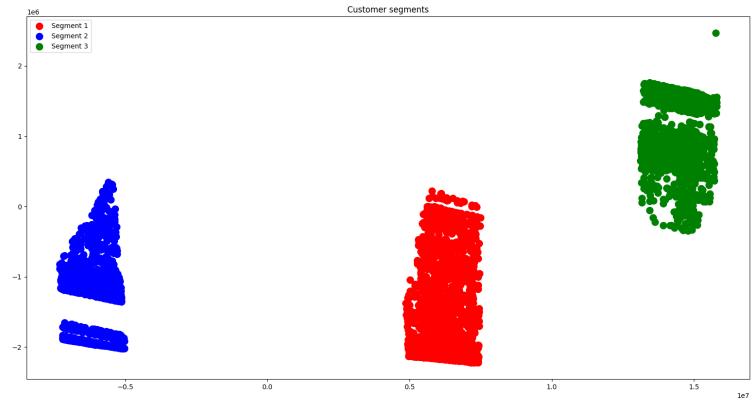


(b) Ukraine

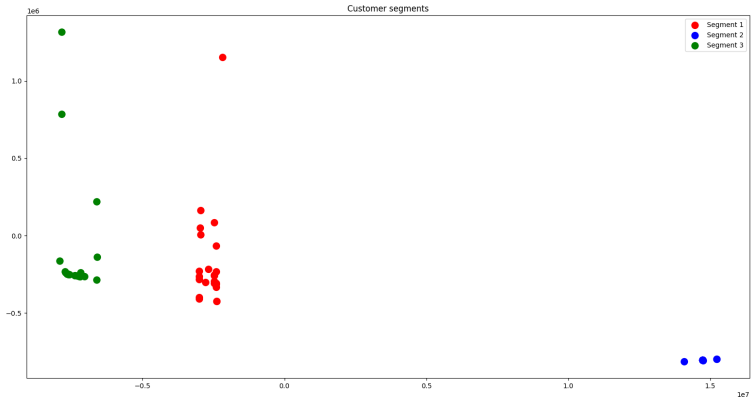


(c) Netherlands

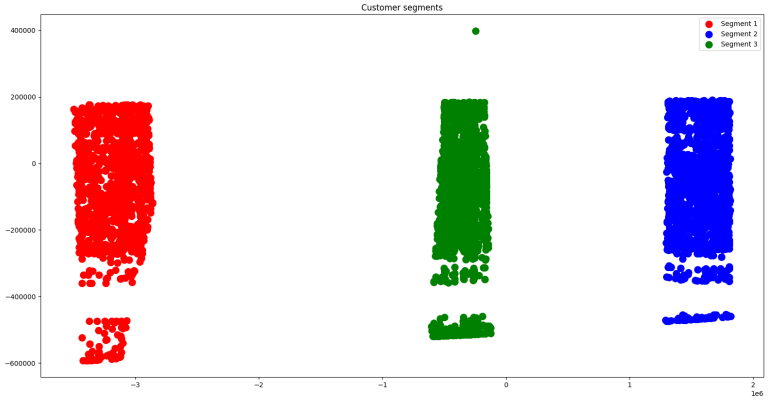
Figure 6.5: Country-wise Customer segmentation based on revenue



(a) Sweden

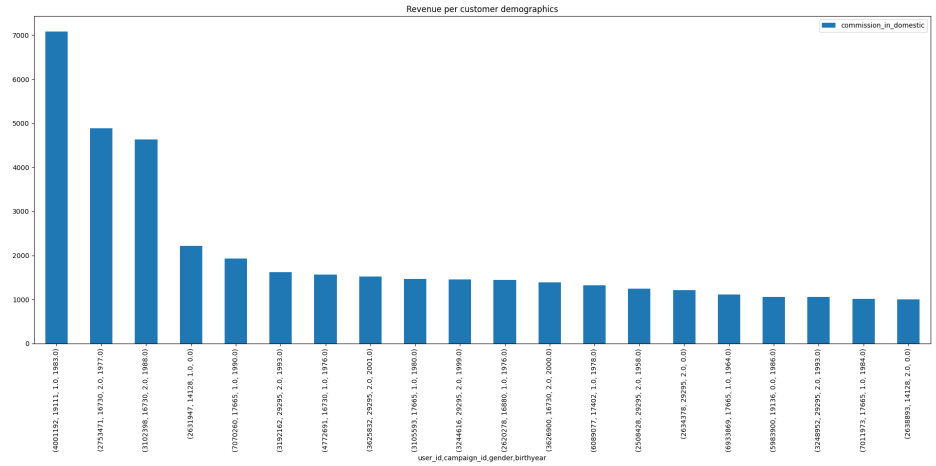


(b) Ukraine

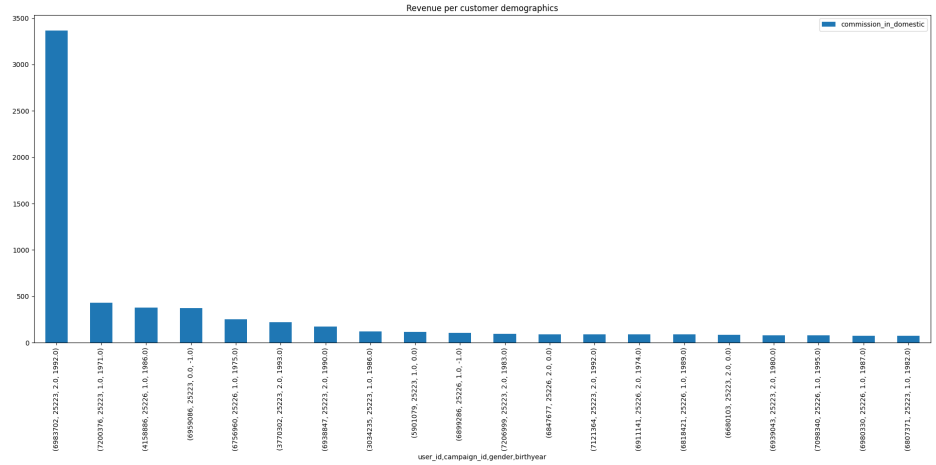


(c) Netherlands

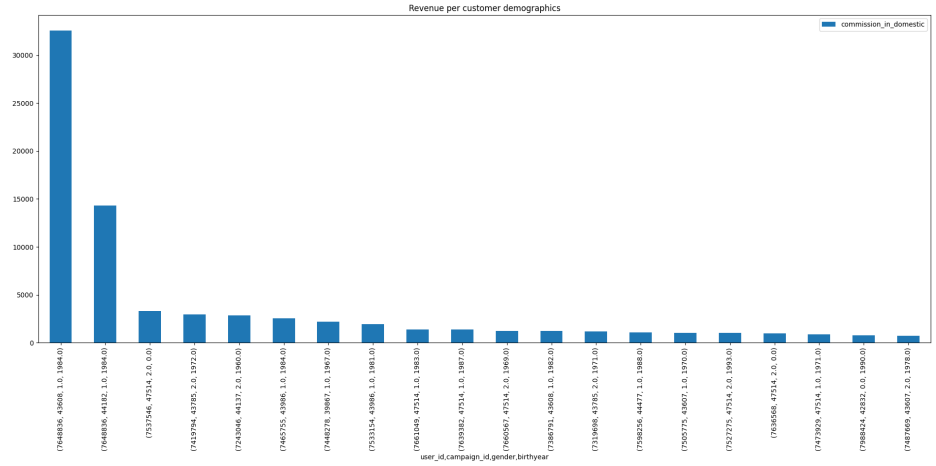
Figure 6.6: Country-wise Customer segmentation based on number of clicks



(a) Sweden



(b) Ukraine

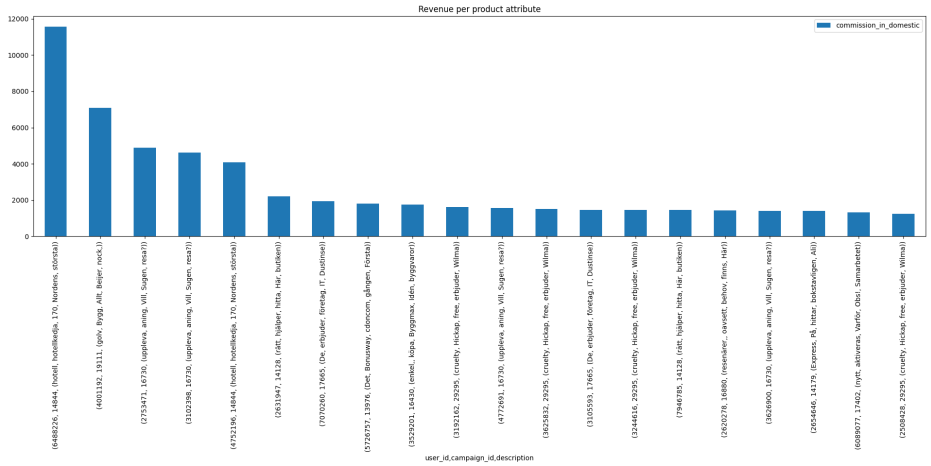


(c) Netherlands

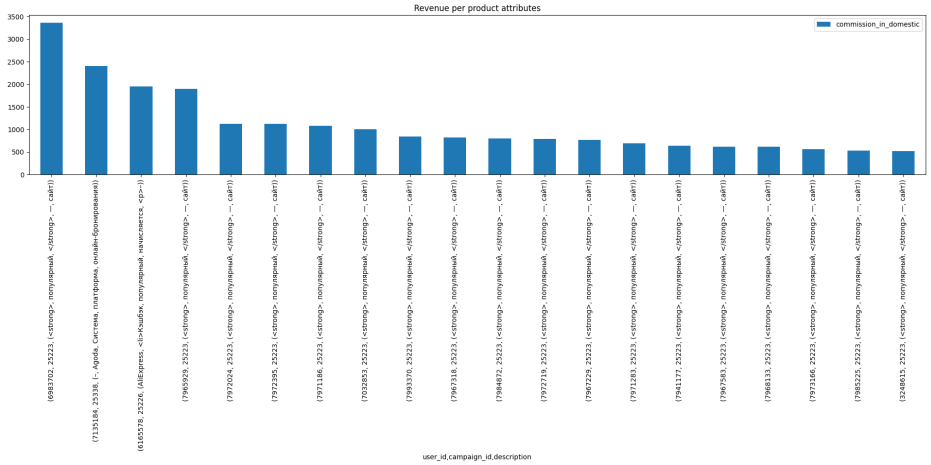
Figure 6.7: Country wise revenue per customer demographics based on gender and age

to the product itself for instance genre in case of books, holiday destination in case of vacations, cuisine in case of food. Based on the figure 6.8, the top20 products dominated revenue generation in customers between 2019 and 2020. Chinese wear is seen in the Netherlands, while in Ukraine products from store 25223 are the top sellers. Sweden shows a mixed trend, but its hotel industry is the one generating the most income.

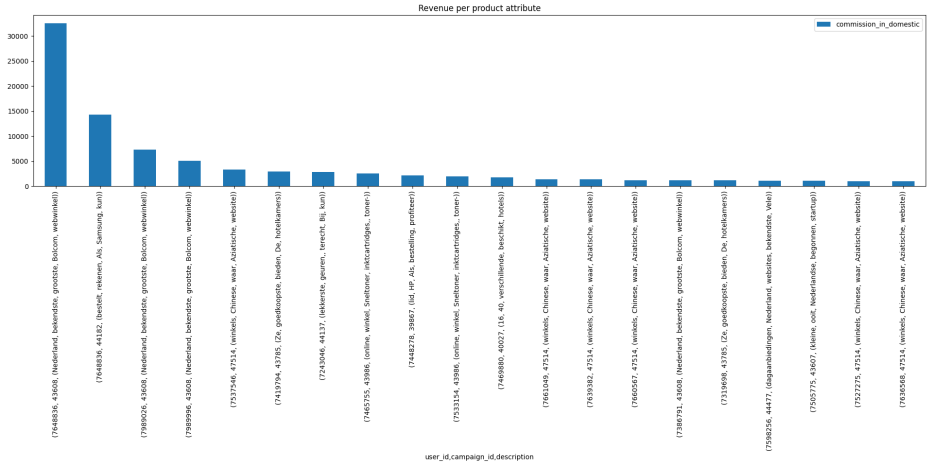
There is a potential side effect of this method in dealing with extremely large datasets that have many dimensions. Both matrix factorization and similarity measures are highly computationally-intensive methods that are used here. It becomes more difficult to learn from similar training features as the dimensionality between observations grows, since predictions for new samples are less likely to be based on previous observations. Due to the exponential growth in the number of features, a generalization is so much more difficult. Increasing the number of dimensions increases the possibility of overfitting to noise, leading to a poor generalization performance. To overcome the challenge of dimension, a better way of modeling data is needed, especially modeling of data in a space of lower dimensions where the relative positions of points in that space convey information about their mutual relationships. That typically refers to transforming discrete activation information into continuous information. Deep neural networks are capable of drawing nonlinear, complex-shaped decision boundaries around points in this vector space. The next chapter will demonstrate how to combine co-training and deep learning methods to overcome the challenges of generalization and dimensions.



(a) Sweden



(b) Ukraine



(c) Netherlands

Figure 6.8: Country-wise revenue per product attributes

Chapter 7

Deep multi-label co-training

In this chapter, we present a multi-labeled semi-supervised deep learning study, which includes a small labeled label set and a set of unlabeled data. As described in the preceding chapters, co-training can be further extended by adding a variety of approaches to enhance the efficiency and processing. In our study, we used multi-label classification, matrix factorization, and similarity measurement techniques to construct a learning approach that used two classifiers for accurate label prediction using a relatively small labeled sample. This method, however, becomes computationally intensive as the amount of data grows. Therefore, this chapter tries to exploit the potential of deep learning when combined with co-training. Through co-training, we implement semi-supervised learning for text classification by implementing two neural networks and a deep learning model to implement a multi-label classification model. This co-training uses two independent classifiers in order to classify the text by dividing the whole corpus into two distinct and mutually exclusive views. Using deep neural networks in this way, the deep co-training model calculates similarities in probability distributions of predicted outcomes by training using different views of the samples. Using this co-training framework, text is classified into multiple labels based on the training of co-trained networks. By training networks together, data can be better analyzed. The database of survivor stories will also be used for the purposes of evaluation so that we can compare every method presented so far with the one being examined in this section. Using

human rights datasets also provides insight into how the human rights domain would be able to benefit from deep learning methodologies and take advantage of its potential if they aligned their data analysis and research.

In their articles Alsharif et al. (2020); Alloghani et al. (2020), analyze and discuss deep learning methods combining supervised and unsupervised techniques to show how these approaches have gained wide acceptance due to the rise of big data and artificial intelligence. Furthermore, this has led to the creation of novel deep learning algorithms that use parallel processing, graphics processing units, and can handle very large datasets. Research has shown that neural networks perform exceptionally well on a range of machine learning tasks, even the most challenging ones, as demonstrated in a comparison of these approaches in Kratsch et al. (2020). However, typically a human, who may need to be an expert, must manually label or annotate the data in order to reach this high performance. Unfortunately, there is a lack of labeled, annotated data required for in-depth research, so existing machine learning algorithms are hard to train effectively. Furthermore, several supervised machine learning algorithms struggle to classify a large corpus of data into multiple labels in an efficient way as stated in this chapter by Pant et al. (2019). The dilemma of choosing between efficiency and accuracy is presenting a challenge.

A very interesting option when constructing a labeled dataset would be to use text-based datasets that can be labeled without human involvement, given the massive amounts of unlabeled text-based data collected by various organizations over the years. In this chapter, we address the semi-supervised text classification problem by combining an unlabeled dataset with a small labeled one in order to create better classifiers that can categorize the unlabeled dataset into one or more labels. In terms of domain of application, we have opted to develop this work in the context of the human rights domain, in which data mining and machine learning have been increasingly applied in the last decade, with several researchers performing analyses of human rights related data.

7.1 Insights from previous chapters

We will be focusing on semi-supervised multi-label text classification to build the framework. In semi-supervised classification, an algorithm is trained to compare unlabeled and labeled instances in both the labeled and unlabeled datasets and to do so in a manner that is better than the supervised algorithm trained on only the labeled instances. It was demonstrated in the previous chapters that employing a hybrid framework, which combines several techniques with co-training, is successful in classifying the unlabeled corpus of text. In order to advance the scope to deep learning, it is necessary to incorporate key insights from the previous chapters. These are:

- **Incorporation of domain knowledge:** Co-training approaches mostly only deliver satisfactory results if the specified working assumptions are satisfied and the training data is large and informative enough. In practice, however, the distribution of the dataset is unknown and does not necessarily meet these ideal conditions. A model's performance will decline when the distributions of labeled and unlabeled data do not match or when the assumptions of the model do not hold. In order to mitigate the performance degradation, we can incorporate richer and more reliable domain knowledge into the model. We saw this while mitigating the problem of the imbalanced labels in chapter 3 and the cold start problem in chapter 5.

- **Unlabeled labels ranking:** An interesting idea was to rank labels for unlabeled samples that are added to the training set with the labeled samples. In chapter 4, it was shown that using matrix factorization, it was possible to label some unlabeled samples based on top-ranked labels derived from a small number of labeled samples. Therefore, extending the labeled set on which the two classifiers will train to classify the remaining unlabeled data.

- **Imbalanced semi-supervised learning:** In real-world applications, the class imbalance is a common issue. When the training data has a high degree of imbalance,

most learning frameworks will display a bias toward the majority, and in some extreme cases may even completely ignore the minority group. This will negatively affect the accuracy of predictive models. The training dataset is generally assumed to be distributed uniformly across all classification labels in order to handle the semi-supervised problem. Chapters 3 and 5 have shown some approaches that successfully addressed this imbalance problem.

- **Exploit the disagreement:** Co-training based on disagreement is intended to train multiple learners and exploit disagreements during the learning process, Zhou and Li (2010). The optimal label is determined by the most confident prediction between two different networks trained simultaneously. We addressed this disagreement in chapter 4 by considering the top ranked labels to be the predicted labels, and in chapter 5 by generating samples with stories that had labels associated with stories with the highest degree of similarity.

- **Distinct views:** A different set of reasonable assumptions leads to a different combination of labels and unlabelled data, and therefore, different algorithms are designed to exploit these combinations. In all previous methods, there are two distinct and mutually independent views, which are aligned to Blum and Mitchell (1998) assumption that instance x has two conditionally independent views, each of which can be classified.

7.2 Deep multi-label Co-training

We present in this chapter the design of a deep neural network, Kihlman and Fasli (2021), that combines a multi-label co-training framework to enhance its capacity to classify a dataset with a very large volume. There are a few assumptions associated with co-training for deep learning, that correspond to Blum's and Mitchell's method. These assumptions are:

- According to Blum and Mitchell (1998) every data x in the dataset has two different

and complementary views, and each view is sufficient to train a good classifier. This assumption leads to co-training using two different classifiers on these two views. Using the two classifiers, we predict each view's unlabeled data and label the most reliable candidates for the other model. The process is iterated until unlabeled data have been exhausted, or until a certain condition has been met (for instance a maximum number of iterations reached). Let v_1 and v_2 be two different views of the same data such that $x = (v_1, v_2)$. This technique assumes that C_1, the classifier trained on View-1 v_1 , and C_2, the classifier trained on View-2 v_2 , make consistent predictions on X. The co-training assumption can be expressed as follows:

$$L_{cot} = H\left(\frac{1}{2}\left(C_1\left(v_1\right) + C_2\left(v_2\right)\right)\right) - \frac{1}{2}\left(H\left(C_1\left(v_1\right)\right) + \left(H\left(C_2\left(v_2\right)\right)\right)\right) \tag{7.1}$$

where

- H is the cross entropy loss for C_1 and C_2
- L_{cot} is total loss for the Co-training assumption
- D is the total dataset
- the co-training assumption is given by

$$C(x) = C_1(v_1) = C_2(v_2), \forall x = (v_1, v_2) \approx D \tag{7.2}$$

- It is the difference and complementary nature of both views that makes Co-training successful. Loss function L_{cot} however only ensures the model makes consistent predictions when applied to the dataset. To address this problem, Qiao et al. (2018) suggests to add the View Difference Constraint to the previous co-training model, and formulated as:

$$\exists D' : C_1(v_1) \neq C_2(v_2), \forall x = (v_1, v_2) \approx D' \tag{7.3}$$

where

– D' are the generated samples from X

In the View Difference Constraint, the main goal is to minimize the cross-entropy between $C_2(x)$ and $C_1(g_2(x))$, where $g()$ denotes the samples generated by any generative model on X.

Based on the above assumptions, we extend the multi-label co-training framework by deploying two neural networks, $F1$ and $F2$ continue to be trained according to Blum and Mitchell (1998), which initially learns a separate classifier for each view on the labeled samples L, and then gradually adds the predictions of the two classifiers on a subset of samples U' from the unlabelled set U to L to complete the training. Moreover, the views on which the networks are deployed should be independent. We will introduce noise to the data to provide contradictory and independent views that would prevent both classifiers from getting influenced. The noise would be in the form of jumbling the text so that it is unintelligible. Additionally, we introduce three losses, whose sum would represent the overall loss associated with the co-training framework. In the same way as in the assumptions, the goal is to minimise this cumulative loss to ensure accurate predictions.

7.2.1 Adding Noise

The major drawback of semi-supervised learning is that the learner tends to get biased and predict incorrect results, reducing the model's efficiency. Moreover, according to Blum, co-training assumes that $p1(x) = p2(x), \forall x \in D$, this means that $D \cap D' = \emptyset$. Clearly, noise must be introduced so that the classifiers will learn to ignore it and be able to make accurate predictions. A method provided by Edunov et al. (2018) is excellent for augmenting a generated sample training corpus, allowing the efficiency of neural network predictions to be improved. This method adds noise to data in three ways -

- words with given probability

- replace words by a filler token with given probability
- swap words up to a certain range

In this way, noise is added to the two views of the dataset, implying that a sample generated for the $F2$ network can lead the $F1$ network in generating incorrect predictions, but not the $F2$. In order to prevent the networks from overlapping, during the training phase, one or both of the networks need to be cross-trained using the noisy samples. This can be achieved by training network $F1$ with samples trained for $F2$ and vice versa. By adding noise to the dataset, both networks are expected to predict the same outcome on their respective views, despite the noise. In other words, over the training phase, the networks will learn to identify noise and not be affected by it as they predict classes.

7.2.2 Cumulative Loss function

- **Multi-label sigmoid cross-entropy (MLCE)** Labeled and unlabeled datasets are represented by L and U', respectively. $D = L * U'$ can be used to represent the entire dataset. By combining two neural networks on independent views $v1(x)$ and $v2(x)$ of $D(x)$, the algorithm aims to classify the unlabeled dataset U' into one or more categories from L. A custom log function is proposed as a way of improving the classifier's performance. The model will provide two sigmoid values as predictions since the output layer contains two output features. The loss values for output and target pairs will therefore be calculated using a sigmoid CrossEntropyLoss().

$$H(y,\hat{y}) = -\frac{1}{M}\sum_{j=1}^{M}[y_j log\hat{y_j} + (1-y_j)log(1-\hat{y_j})] \tag{7.4}$$

where

  - M is the number of classes
  - z is the output from the neural network

- $\hat{y}$ are the predicted probabilites calculated by applying the sigmoid function σ on z

$$\hat{y_j} = \sigma(z_j) = \frac{1}{1+e^{-z_j}} for score\, z_j \tag{7.5}$$

Based on the number of classes in the output and target pairs, loss values are calculated for each class in L. Seven classes are present in this dataset, so seven loss values are calculated for every pair of output and target. A final loss is determined by averaging out all these losses.

$$L_{mlce}(x,y) = H(y, f_1(v_1(x)) + H(y, f_2(v_2(x)) \tag{7.6}$$

where

- H is the sigmoid cross entropy
- x, y data from L
- $f_1 v_1(x)$ Network on view1 of x in D
- $f_2 v_2(x)$ Network on view2 of x in D

- **Noise sigmoid cross-entropy** The multi-label cross-entropy loss is further extended to incorporate the noisy samples generated for the two independent views. One network is trained by the noise created by the other view, as mentioned previously. Hence, minimizing the cross-entropy between both these networks will aid the training process to not be adversely affected by the additional noise, as well as regularize the model by tightening the decision boundaries while making predictions.

$$L_{noise}(x) = H(v_1(x), v_2(noise_1(x))) + H(v_2(x), v_1(noise_2(x))) \tag{7.7}$$

where

- H is the sigmoid cross entropy

- $noise_1$ is noise generated for $v1$
- $noise_2$ is noise generate for $v2$
- v_1 is view1 of D
- v_2 is view2 of D

- **Kullback-Leibler divergence** The co-training model assumes that for the distribution X from which x is drawn, $F_1(v1(x))$ and $F_2(v2(x))$ agree on their predictions. Therefore, both network $F1$ and $F2$ should have similar predictions on U'. In order to calculate the symmetric divergences between $p1(x)$ and $p2(x)$, the Kullback-Leibler divergence, a measure of similarity, is computed.

$$D_{KL}(g||f) = H(g,f) - H(g) = -(\sum_{j=1}^{M} y_j log(\hat{y}) - \sum_{j=1}^{M} y_j log(y)) \tag{7.8}$$

where

- H is the sigmoid cross entropy
- g is the distribution without noise
- f is the distribution with noise
- y is predicted probabilities for gabel is determined by the most confident prediction between two different networks trained simultaneously. We addressed this disagreement in chapter 4 by considering the top ranked labels to be the predicted labels, and in chapter 5 by generating samples with stories that had labels associated with stories with the highest degree of similarity.
- **Distinct views:** A different set of reasonable assumptions leads to a different combination of labels and unlabelled data, and therefore, different algorithms are designed to exploit these combinations. In all previous methods, there are two distinct and mutually independent views, which are aligned to Blum and

Mitchell (1998) assumption that instance x has two conditionally independent views, each of which can be classified.

- $\hat{y}$ is predicted probabilities for f

According to Sankaran et al. (2016) Kullback-Leibler divergence is a scoring of how one distribution differs from another, where calculating the divergence for distributions p and q would give a different score from q and p. It is essential that both noise distributions on which the models are learned are very similar to the actual distribution so that the KL divergence is as small as possible. This can be achieved by minimizing KL divergence loss. Therefore the KL loss for both the networks will be calculated as

$$D_{KL} = (H(g_1, f_2) - H(g_1) + H(g_2, f_1) - H(g_1)) \tag{7.9}$$

where

- H is the sigmoid cross entropy
- g_1 is the distribution with out noise for network1
- f_1 is the distribution with noise for network2
- g_2 is the distribution with out noise for network2
- f_2 is the distribution with noise for network2

7.2.3 Model

In this multi-label neural network model, there are 4 linear layers for the learning model, with the first layer starting with 50 features (the features extracted from the labelled set L by TF_IDF) and ending with 256 features for the final layer. For each label (7 in this case), there is an output feature that produces a two feature sigmoid output (0 and 1). Of the two

sigmoid output values, the one with the higher value will be selected as the predicted output for that label.

7.2.4 Loss function

By taking the linear addition of the multi-label Entropy loss, the Divergence loss, and Noise loss, the model can be optimized. To further improve prediction accuracy and minimize the loss complexity, tuning the learning rate and applying L2 regularization on the divergence and noise would result in improving the predictions.

7.2.5 Training

For each iteration, data in the form of $[d_l1, d'_u1]$ and $[dl2, du'2]$ created from the two independent views is fed to the respective neural networks involved for the training. d_l is the set of labeled samples and d_u is the set of unlabeled samples. From these noisy samples x_{noisy1} and x_{noisy2} are generated and crossed fed to the neural network. This means x_{noisy2} is fed to Net1 and x_{noisy1} is fed to Net2.

This process is repeated many times employing gradient descent with a declining learning rate until all the unlabeled samples get labeled.

The proposed algorithm is shown in figure 5

7.3 Experimental setup

In this dataset, 9000 stories about Xinjiang victims are collected. This dataset is publicly available for download on Kaggle and on their website,

The data has been pre-processed to make it suitable for classification. As part of the normalization process, stop words and punctuation are removed from the text, and all numbers are rounded to the nearest whole number. Additionally, the top 50 features were generated with the TF_IDF vectoriser. A total of 7 labels have been manually annotated on this data,

Given:
a set L **and** U' **of labelled and unlabeled samples**
Divide training set D **into** $v1$ **and** $v2$
Create a pool $p1(xv1, yv1)$ **for** $v1$ **and** $p2(xv2, yv2)$ **for** $v2$ **from** L
Create a pool by choosing u **at random from** U'
for $i \leftarrow 0$ **to** k **by** 1 **do**
  Generate noisy sample x_{noisy1} **for** $v1$ **where** $\forall x \in p1 \cup u$
  Generate noisy sample x_{noisy2} **for** $v2$ **where** $\forall x \in p2 \cup u$
  Use $p1$ **to train** $h1$ **on the** x_{noisy2} **portion of** D
  Use $p2$ **to train** $h2$ **on the** x_{noisy1} **portion of** D
  Calculate multi-label cross entropy loss
  Calculate noise loss
  Calculated divergence loss
  Final loss = $loss_{mlce}$ **+** $loss_{noise}$ **+** $loss_{kl}$
  Compute the gradients with respect to the final loss and update $v1$ **and** $v2$
  Continue until U' **is exhausted**
end

Algorithm 5: Proposed co-training Algorithm

and each story is further classified according to one or more of these labels. For the loss to be stable across different training iterations, each data pool must have the labeled set proportional to the threshold set for the experiment. To accomplish this, the labeled and unlabeled data is evenly divided as indicated in the algorithm to build each data pool.

7.3.1 Evaluation

The performance of this model must be thoroughly evaluated by comparing it with state-of-the-art multi-label classifiers using the OneVsRest supervised learning method, the extensions of the co-training model, and the deep multi-label classification model. The accuracy of these classifiers, in combination with their error rates, serves as the metric for evaluation. In the equation, accuracy shows how many correct predictions were made, whereas the error rate shows how many errors were made for each decision. The two factors must be considered together when evaluating the performance of a classification model.

7.3.1.1 OneVsRest

OneVsRest uses a binary mask to compare multiple labels in a multi-label algorithm. Each prediction will yield a set of 0s and 1s representing the class labels applicable to each row. The OneVsRest strategy can be applied to multi-label learning, which uses a classifier to predict multiple labels. All classifiers supporting multi-class classification, including SVM, Naive Bayes, and Random Forest, can be wrapped in the OneVsRestClassifier to predict multiple labels.

7.3.1.2 Co-training Extensions

In this method, using co-training classifiers like SVM, Naive Bayes or Decision Trees are deployed over the dataset. The classifiers predict the labels for the stories based on co-training, matrix factorization, and recommendation.

7.3.1.3 Deep Multi-label classification

A neural network can support multi-label classification directly by simply specifying the number of target labels in the problem as the number of nodes in the output layer. Sigmoid activation must be used for each node on the output layer. This will predict the probability of the label belonging to one class and this way predict labels for all.

7.4 Results

The experimental results presented here are for co-training performed on the victims' dataset and compared with several supervised state-of-the-art classification algorithms, extensions of the co-training algorithm, and deep multi-label learning.

- **Supervised classification algorithms** Algorithms like Decision Trees, K-nearest neighbour, SVM, and Naive Bayes are the most common supervised learning algorithms used to compare results. A multi-label supervised learning scenario has shown these

algorithms to be the most effective. To report the results of supervised learning, a 10% threshold is kept for the labeled set.

- **Co-training Extensions** Multiple co-training extensions have been proposed by utilizing Blum's co-training principle. Co-training with matrix factorization, multi-label co-training, and co-training with similarity recommendations are the ones used for this experiment. The experiment employs 10% of the labels in the set to predict the results.

- **Deep multi-label learning** The experiments conducted before focused on simple methods of supervised or co-training. The proposed algorithm being a deep learning algorithm, its results should be compared to the current state of the art in deep learning. The classification results are constructed with 10% threshold for the labeled sets based on a simple multi-label deep learning model.

As seen in table 7.1 the performance of the different algorithms used for this experimental setup is compared. According to our results, co-training with recommendations is the most effective, followed by deep multi-label co-training. Even though co-training with recommendations shows better results, multi-label training is still more productive in terms of time complexity and training. Firstly, co-training with recommendations ranks labels based on matrix factorization and then predicts new labels based on new results from that. Next, similarity measures are used to boost the confidence in the newly predicted samples. Therefore, with increasing thresholds of the labeled set, the time it takes to generate a prediction exponentially increases. In contrast, deep multi-label co-training takes only a short period of time to train and predict new labels despite the fact that it adds noise and uses two neural networks. The time complexity and training time remain the same regardless of how high the labeling threshold or how much noise is added as shown in table 7.2. In this table, it can be seen that adding in more labels helps the network to identify noise and not get influenced by it. This table also displays results for a fully supervised multi-label classification along

Supervised multi-label		
classifier	accuracy	error rate
Decision Tree	0.710	0.279
KNN	0.536	0.463
SVM	0.714	0.265
Naive Bayes	0.710	0.289
Co-training multi-label		
Decision Tree	0.432	0.567
KNN	4.901E+016	0.509
SVM	0.499	0.500
Naive Bayes	4.93E+016	0.507
Co-training multi-label with matrix-factorization		
Decision Tree	0.624	0.375
KNN	0.705	0.394
SVM	0.695	0.304
Naive Bayes	0.730	0.269
Co-training multi-label with similarity recommendations		
Decision Tree	**0.766**	0.233
KNN	0.6299807	0.370
SVM	0.756	0.243
Naive Bayes	0.723	0.276
Deep multi-label		
NN	0.724	0.275
Deep multi-label co-training		
NN	**0.726**	0.273

Table 7.1: Classifier comparison on 10% of the labeled samples

with co-training extensions. Here it can be seen that in spite of using fewer labels than its fully supervised counterpart, multi-label co-training showed superior results. It aligns with the purpose of this chapter to be able to efficiently train a model with minimal human involvement.

Threshold	accuracy	error rate
10%	0.726	0.273
20%	0.735	0.264
30%	0.760	0.239
Deep multi-label at 100% labelled set		
NN	**0.808**	0.191

Table 7.2: Deep multi-label co-training comparison on various threshold of the labeled samples and fully supervised deep learning

7.4.1 Comparison of loss methods

Combining custom losses helps improve the performance of the model as well as preventing the two neural networks from getting influenced by each other.As discussed in the section, this model has three loss functions that contribute to its performance and efficiency. The three losses are the cross-entropy, noise, and divergence losses of a multi-label dataset. A performance analysis of the algorithm is shown here by removing some custom loss to focus on the importance and effect of minimising all the three loss functions simultaneously during training. This experiment compares the performance of the model by plotting the accuracy at each epoch when one or more of the losses have been removed during the training phase. Furthermore, it is also tested at various thresholds of using the labeled data while training the network. Iterations over 100 epochs are used in the training.

As observed in table 7.3, applying a loss function on its own resulted in the deterioration of prediction accuracy. Although adjusting the threshold of the labeled samples helped improve the performance a bit, it was not as significant as when all three loss functions were applied together. It is also observed that when just noise and divergence loss are combined, each network tends to predict randomly. Furthermore, increasing the number of epochs shows that the model starts to over-fit. The efficiency of the model does not seem to be substantially impacted by minimizing these loss functions.

The average accuracy per epoch shows that if all three loss functions are linearly combined, the accuracy increases steadily until between 55 and 60 epochs, it suddenly drops.

This could be caused by networks being slightly impacted by noise. In the end, the model maintains a steady rate, picking up accuracy, until all epochs are exhausted. However, comparing the performance of using only divergence loss shows that when the noise is added to the model, the model gradually starts to identify it and grows to oversee it for future predictions. Furthermore, the performance deteriorates severely when only considering the noise loss without divergence, because, from epoch 45 onward, the networks appear to collapse into each other. Finally, without the MLCE loss, the model behaves in an unpredictable manner resulting in a severe decrease in accuracy. During this process, the network is unable to differentiate between real data and noise, causing it to predict randomly once again.

7.4.2 Comparison without noise

Co-training is a procedure in which two neural networks are trained in parallel to analyze discrete views of a given dataset in this thesis. This framework assumes that despite receiving different types of data in the training loop, both of these networks would produce similar results every time. This method, however, has a significant drawback: there is no guarantee that the views provided by the two networks provide different and complementary information about each data point. It is clearly evident from figure 7.3b that although both networks are trained independently for independent views, they end up collapsing into each other and giving random predictions. The result of this can be identical networks and the model cannot benefit from them. Noise is therefore added to the dataset in order to mitigate its influence over the training process so that it can be identified by the two networks and not to affect their performance. From the figure, it is evident that adding noise confuses both networks and pushes them away from each other, though they predict different results. It supports the argument made at the beginning of the chapter that by adding noise we can ensure that the networks are able to detect noise and not be impacted by it during the training process. It has been shown that both the networks have different views and over time they gradually start to converge, overlooking noise and making good predictions.

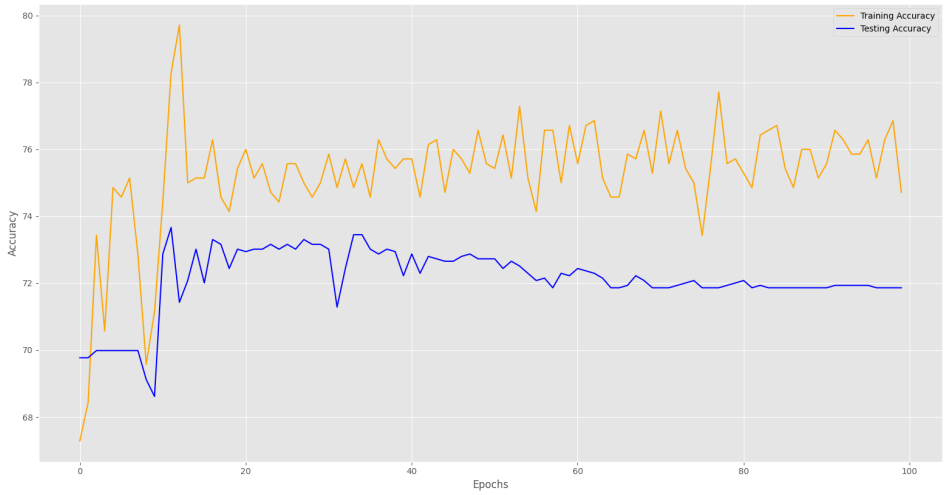
MLCE loss comparison		
Threshold	accuracy	error rate
10%	0.710	0.298
20%	0.727	0.272
30%	0.723	0.276
Noise loss comparison		
Threshold	accuracy	error rate
10%	0.699	0.3
20%	0.728	0.225
30%	0.743	0.256
Divergence loss comparison		
Threshold	accuracy	error rate
10%	0.706	0.293
20%	0.714	0.285
30%	0.743	0.257
Divergence loss and noise comparison		
Threshold	accuracy	error rate
10%	0.533	0.3
20%	0.538	0.3
30%	0.536	0.3

Table 7.3: loss comparison on various threshold of the labeled samples

7.5 Conclusions

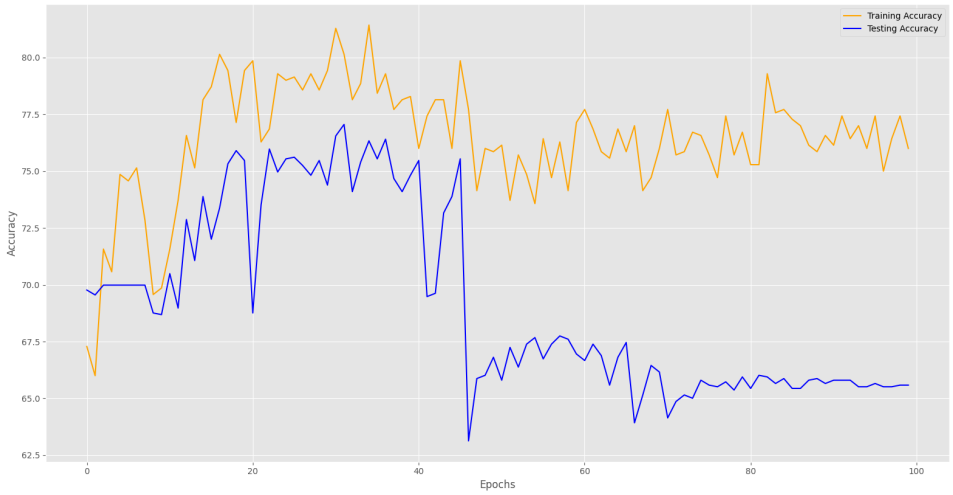
Utilizing a deep co-training model designed for multiple labels was presented in this chapter. Developing this model was prompted by the need to classify very large volumes of unlabelled data when only a small set of training data is available. The novel aspects of this method include three new losses based on three primary objectives - multi-label, noise and convergence of neural networks. We also generated noise samples from the unlabelled corpus in order to confuse the classifiers by overriding the noise and forcing them to learn from the actual data. This work extends the co-training learning method by deploying two independent neural networks. Based on the assumption that the data has two independent views, it is assumed that two effective classifiers can be built on these views and will accurately predict the unlabelled text. In light of the recent successes of deep neural networks in supervised learning, this framework leverages them to apply deep networks to the problem of

(a) Accuracy with all three loss functions

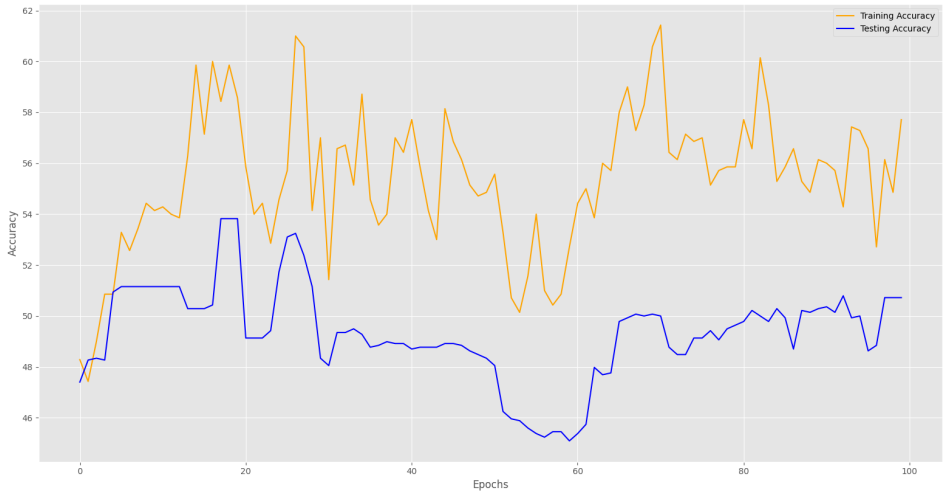


(b) Accuracy with just divergence loss

multi-label text classification, where data can be classified into another label at a later point. The experiments conducted revealed that it was extremely difficult to create independent views so that the networks predicted the same results. This was prevented by adding noise to the data and creating samples that were then cross-trained with either of the networks. In this way, the views were discrete and their predictions were distinct. It was further demonstrated that by using three different loss functions, the model training was improved and both



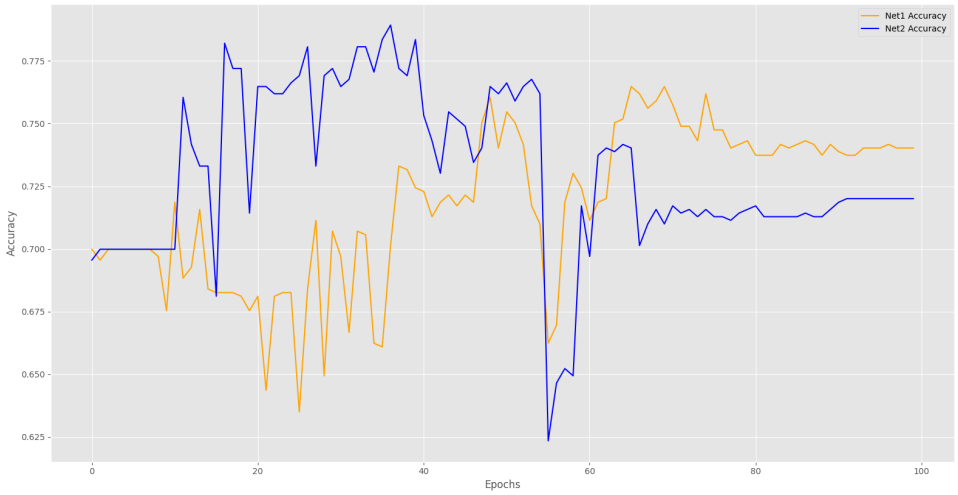
(a) Accuracy with just noise loss



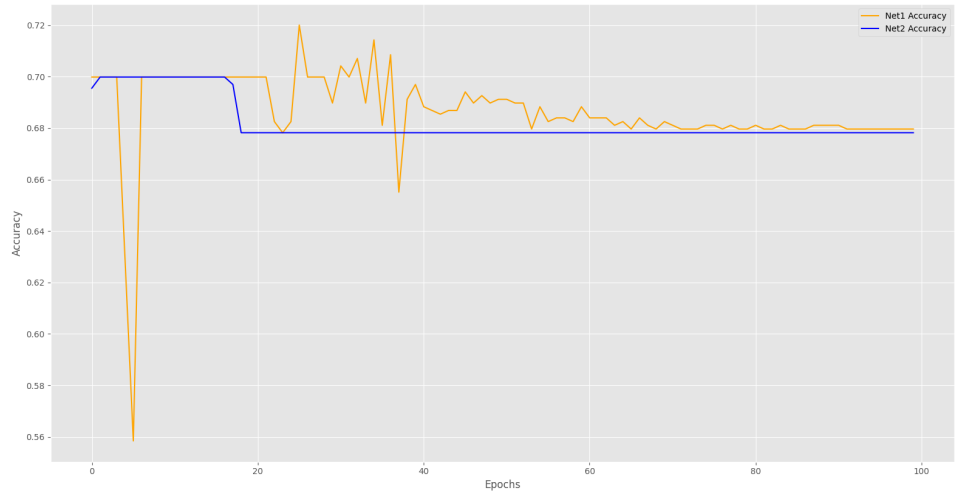
(b) Accuracy with noise and divergence loss

Figure 7.2: Loss effect on Test/Train accuracy at 30% labelled set

networks were prevented from getting influenced by the added noise. In comparison to its state-of-the-art supervised counterparts, the model performed better with a smaller labelled sample than with an entirely supervised model. In contrast, the model's performance was slightly lower than the model with recommendations when using co-training. The model, however, tended to become more complex as the number of data samples increased. Despite



(a) Network performance with noise



(b) Network performance without noise

Figure 7.3: Effect of noise on model performance

the growth of both labelled and unlabelled samples, the model presented in this chapter still remains efficient as compared to using 100% labelled set.

The present study used the domain of human rights violations as the study domain for experimenting with co-training with deep learning for text classification. Results of the experiment show that this method can classify a large corpus of victim stories into categories

that the human rights experts identified as important in this situation. In addition, the method demonstrated that the networks classified correctly even when adding noise. One could view this as a way forward when survivor stories, for example, or content about human rights violations could get tampered with with irrelevant information resulting in misinformation and incorrect classification. Aiming to identify and ignore such forms of misinformation, this chapter attempts to identify and eliminate them. As demonstrated in this chapter, the method requires an initial small labelled set in order to work, and because of that, it is not confined to any particular domain. It can therefore be applied to domains where experts can provide a minimal labelled sample set.

Chapter 8

Conclusions and future work

In this thesis, we investigated the possibility of building a model that could leverage a small annotated dataset to classify and label a dataset into one or more categories. Co-training is a semi-supervised learning approach that trains two classifiers based on two different views of the data. It assumes that each sample is described using two different feature views that provide a complementary view of the sample. Tom Mitchell and Avrim Blum introduced it in 1998. We use human rights violations as the domain of interest, such that the survivor stories could be categorized into one or more violations by using a labeled set annotated by experts in this area. By introducing additional techniques that would enhance the accuracy and efficiency of the model, this thesis aims to maximize the potential of Blum's co-training method. This thesis addresses the following:

- Finding a way to increase the number of labeled samples in the training set so that the training set has enough labels to accurately identify the different documents.
- Provide at least one or more labels for each document.
- Rank each label based on its significance among the annotated data.
- Using the previously classified stories, recommend labels that are similar for the unlabeled document.

- Make a model that is less computer-intensive yet accurate.

Through Chapter 3, Chapter 4, Chapter 5, and Chapter 6, we introduced the methods that evolved toward answering the questions that made up the backbone of this research. We used these methods to categorize textual data into appropriate categories over both experimentation and real data. This work has a strong focus on human rights as a domain but also extends to other domains for instance finances and online shopping.

In Chapter 3, we extend the already useful semi-supervised co-training method to allow multiple labels to be assigned. With this change, Blum's method now had one or multiple labels associated with a story or document. The purpose of the case study was to predict new labels for a small section of the dataset based on a limited dataset of labels and then train using these new labels to classify the remaining data. The model also looked at how to overcome class imbalances by using random sampling to choose more samples from one class than another in order to compensate for an imbalance that is already present or likely to emerge if a random sample was selected. As well as implementing a confidence score, we calculated the confidence interval that is used to determine the classification error and accuracy. As a result, we established that the lower the confidence interval, the better the prediction.

In Chapter 4, we analyzed the reasons behind the attribution of particular labels to stories. Two new approaches, matrix factorization and singular value decomposition (SVD), are introduced so that hidden and less obvious relationships between the labeling of the victim stories can be determined. This model aimed to reveal underlying patterns between stories with similar labels and to determine the relationship between the labels and the stories. We developed a latent feature model using SVD and utilized matrix factorization to compute additional labels from the labeled data, in order to identify violations using the latent factor model. Furthermore, a recommender system was developed to predict the stories that will be similar to those predicted by the label.

After discussing matrix factorization and recommender systems, chapter 5 explores

methods to extend the framework to provide recommendations that provide feedback or ratings when no historical data is available. The classifiers of any recommendation system need enough data to be able to determine a user-item relationship, or in the case of this thesis' experimental dataset of survivor stories, a label-story relationship. This model attempts to solve the cold-start problem by using several similarity-based algorithms, including Euclidean distance, correlation, cosine angle, and Manhattan distance, as well as content filtering. When stories are similar in nature, either by content or semantics, relationships are established. The similarity is particularly useful when insufficient information is available to construct the story/label relationship matrix for the survivor stories database used to evaluate the model. This model demonstrated that it could increase the likelihood of a new story being correctly labeled.

Through chapter 6, the framework was extended to take advantage of neural networks by deploying them in a co-training framework for multi-label classification. When only a small set of training data is available, the need to classify very large volumes of unlabeled data was a motivating factor for developing this model. To identify additional hidden relationships between the stories and the labels, this model used two neural network classifiers to train simultaneously on the same labeled data set. Furthermore, the dataset was also combined with noise so that both neural network classifiers would recognize it as noise and not get affected by it during classification. As a result, the stories were accurately classified into one or multiple labels. A cumulative loss function was also found to successfully optimize the parameters for the two neural network classifiers in order to achieve accurate prediction.

8.1 Grounding the Contributions

As explained and demonstrated by Blum, Co-training is a semi-supervised training method in which the dataset is divided into two views that are independent and conditionally discrete. Each view is classified independently by two classifiers. The probability confidence

value is checked after training against a predefined threshold for both views. View 2's high-confidence values are added to View 1's training set, and vice versa, thereby training and teaching one another. In accordance with Zhu and Goldberg (2009), co-training makes two major assumptions. In the first assumption, all views are conditionally independent of class labels. Otherwise, artificial views are created. Such an assumption is known as the independence assumption. According to the second assumption, each view can adequately be labeled independently when the data is sufficient. This assumption is known as the sufficiency assumption. The process is gradual and takes time. Co-training will work only if the two assumptions are satisfied. Real-world scenarios may however only partially or not entirely meet these criteria. The views must, therefore, be carefully chosen or some method must be required to meet the assumptions. Blum's preliminary results in classifying web pages demonstrated that this method could be beneficial, but that it requires further study. The COTRADE model, Zhang and Zhou (2011) is further improved by presenting Confident cO-Training with Data Editing, which improves the quality and reliability of communication between views. With the knowledge that co-training can be extended, we demonstrates some methods in the previous chapters combining co-training and how that compares with state-of-the-art classifiers. We will go into further detail about the contributions made by this thesis in the sections that follow.

8.1.1 Multi-label co-training for text classification

Multi-label text classification is the idea driving this extension. As for survivor stories, it would be interesting to know how many violations did the victims encounter to ensure that they received the best rehabilitation service. Thus, we utilize a semi-supervised learning method with a co-training approach to solve multi-label learning problems, effectively exploiting abundant unlabeled data to improve text classification performance. The main contributions of this method are summarized below.

- We have extended the single-label co-training algorithm to the multi-label algorithm

following Blum's co-training principles: (i) features within a dataset can be divided into two groups; (ii) each group of sub-features can be trained to produce good learning; and (iii) the groups are conditionally independent.

- It addresses class imbalance. It is rare to find an exact equal number of instances in every classification in most classification data sets. As a result, classifiers are biased towards the class that has the most labels and predict the same class repeatedly. Using re-sampling techniques that over-sampled minorities and under-sampled majorities, this method avoided this problem.

- It addresses prediction confidence. The method calculates the percentage of classification errors made by the classifiers using this approach. It is taken into account when labeling the set that has the lowest percentage as the most confident prediction.

On the basis of experimental results on the survivor stories dataset, this method outperforms SVC (linear) and linearSVC algorithms in terms of accuracy. The confusion matrices were used to illustrate how the label imbalance among the initial labels was overcome in the extended version of the multi-label co-training. Further, it was shown that this method produced better results by only training on 10% of the labeled set as compared to other methods that relied on almost 70%. While this method does appeared to be effective, there were some concerns on why the classifiers predicted a certain label and their consistency in predicting the same label. This poses a critical concern, since the method may misclassify. A misclassification of survivors stories can result in incorrect rehabilitation treatments and interventions for torture victims when it comes to human rights cases.

8.1.2 Multi-label co-training with matrix factorization

From chapter 3, we began to investigate why certain labels are grouped into certain clusters and if there were any hidden features that could influence the learning process. We demonstrated in chapter 4 how matrix factorization can be used to determine hidden and

less obvious connections between the human rights experts' labels given to surviving victim stories. According to this model, both items and users (using these stories as examples) can be distinguished based on patterns and trends. Both users and items are mapped into a joint latent factor space of dimensionality that represents the degree of association between an item (label) and its features. Hence, the neighborhood group of labels in the labeled dataset is built.

The main contributions of this method are summarized below.

- Collaborative filtering based on matrix factorization -The basic principle of this model is that both items and users (using these stories as examples) can be distinguished by feature vectors based on patterns and trends. Additionally, a user's rating can be linked to a particular item (a label). Using the ratings from the unlabeled set, we can now predict ratings for new stories based on ratings from the remaining stories. By placing the stories and their labels in a two-dimensional matrix, it is possible to create a vector of stories that reveals the missing data.

- Ranking - Based on the annotated information of the experts, SVD enabled us to rank the labels. With SVD, a conceptual match may be found Ogheneovo and Japheth (2016) which expands the number of suggestions as well as their ranking. By using SVD, we had the ability to map terms globally or conceptually without explicitly connecting them and build a ranking matrix.

- Recommending - Using SVD and cosine similarity, we can predict the labels based on similarities between stories. In addition to handling issues of scalability and sparsity, SVD can minimize negative correlation due to lack of data.

In our experiments, we found that the improved co-training algorithm displays a much lower overall coverage error rate than its relevant counterparts. It can be seen when the predicted labels (one or more) of a story exactly match its actual labels. When used as the base classifier for this model, the GaussianNB classifier performs significantly better than

the others. A GaussianNB classifier recommended with an accuracy rate of over 74%.

This method, however, ignores the cold-start problem. The reason is that all ranking and factorization functions would be limited if historical data were unavailable. Without this information, the algorithm would have difficulty producing these matrices. With stronger similarity measures, it would be possible to discover how patterns form between labels and existing stories when enough data is not available.

8.1.3 Multi-label co-training with similarity-based recommendations

This matrix factorization approach ranks the stories based on their labels, as shown in chapter 4. However, when there is no or insufficient historical data, it failed to provide positive results. To mitigate the cold-start issue of insufficient data, we investigated adding similarity-based recommendations to the co-training algorithm in chapter 5. Stories were compared when they were similar in nature, whether by content or semantics. In the survivor stories database used to evaluate the model, the similarity between stories and labels is particularly useful in cases where there is inadequate information to create a story/label relationship matrix. As a result, fewer labeled samples were required to generate the relationship matrix using this method. The probability of confident samples predicted by this method demonstrated higher precision than that of standard classifiers when co-trained with similarity-based recommendations.

The main contributions of this method are summarized below.

- Similarity Score - We introduced a similarity score to determine how closely two samples are related. It is the inverse of the dissimilarity measure, which indicates how distinct data objects are. For document similarity calculations, Euclidean, Manhattan, and Cosine Distance Measures were used.

- Label selection - We calculate a weighted average of the n-most similar documents based on the similarity score for each label. After that, all labels with a score higher

than 0.5 are added to the existing set of labels as positive labels to the current document.

When deployed on the Bonusway dataset, the algorithm revealed several interesting relationships, they are:

- Usage data and purchase history is useful for determining how engaged the user is with the product.
- Demographic information helps to determine user preferences.
- Product attribute data helps to determine what products users prefer.

Using this method may result in side-effects when dealing with large datasets that have many dimensions. The methods here, matrix factorization and similarity measures, require a lot of computational power. With growing dimensionality between observations, it becomes increasingly difficult to learn from similar training features, as new observations are less likely to be based on previous observations. The exponential growth of features makes generalization so much more challenging. Overfitting due to noise becomes more likely when there are more dimensions, which results in poor generalization. A better way of modeling data is required to overcome the dimension challenge, especially when modeling data in a space of lower dimensions where the relative positions of points convey information about their relations.

8.1.4 Deep multi-label co-training

Using multi-label classification, matrix factorization, and similarity measurement techniques, we constructed a learning system that uses two classifiers for accurate label prediction using a relatively small labeled sample. As the amount of data increases, however, this method becomes computationally intensive. As a result, chapter 6 combines deep learning and co-training. By using deep neural networks in this way, the deep co-training model calculates

similarities in probability distributions of predicted outcomes by training using multiple views of the data. By using two neural networks, the multi-label co-training framework is further trained according to Blum's algorithm in Blum and Mitchell (1998). In order to prevent both classifiers from getting influenced, noise was introduced to the data to provide contradictory and independent views. Noise would be created by jumbling the text, making it unintelligible. A total of three loss functions were developed, whose sum represents the overall loss associated with the co-training framework. To ensure accuracy, minimising this cumulative loss is the goal to achieve accurate predictions. The main contributions of this method are summarized below.

- Adding Noise - In both views of the dataset, noise was added, indicating that a sample generated for one network could lead to incorrect predictions for the other. A cross-training of one or both networks using noisy samples is necessary to prevent overlapping during training. This was achieved by training one network with samples from the other, and vice versa. Due to the noise in the dataset, both networks should predict the same result, regardless of the noise. As they predict classes, the networks will be able to identify noise and not be affected by it over the training phase.

- Cumulative loss - The multi-label cross-entropy loss is calculated as the sigmoid cross-entropy for the output and target pairs. These losses are averaged to derive a final loss. Noise loss is calculated as the sum of the sigmoid cross-entropy of the noise applied to each neural network classifier. By calculating the divergence between the distributions with and without noise, one can score the difference between two distributions. This is the Kullback-Leibler divergence loss. The model can be optimized by taking the linear addition of the Multi-label Entropy loss, the Divergence loss, and the Noise loss. By doing so, prediction accuracy can be further improved and loss complexity minimized.

The experimental results were conducted on the victims' dataset and compared with several supervised existing state-of-the-art classification algorithms, extensions of the co-training

algorithm developed in the previous chapters, and deep multi-label learning. The classification results are constructed with 10% threshold for the labeled sets based on a simple multi-label deep learning model.

The results were evaluated on three factors. They are -

- Classifier performance - The most effective co-training method, according to our results, is co-training with recommendations followed by deep multi-label co-training. Despite better results from co-training with recommendations, multi-label training is still more productive in terms of time complexity and training.

- Effect of the custom loss functions (all and individual) - By removing some custom loss functions from the algorithm, we were able to focus on the importance and effect of minimising each loss function individually. As part of this experiment, the accuracy at each epoch is plotted when one or more of the losses have been removed during training to compare the model's performance. Additionally, it is also tested at different thresholds of using labeled data when training the network. The model was found to benefit from minimizing the cumulative loss function and to predict accurate results.

- The impact of noise - Despite predicting different outcomes, it is evident that adding noise confuses both networks and pushes them apart. As compared to the model that did not have noise added, it showed that both networks had different views and over time, they gradually started overlapping, overlooking noise and making good predictions.

The model outperformed its state-of-the-art supervised counterparts even with a small labeled set. Despite the growth of both labeled and unlabeled samples, the model presented in this chapter is still effective when compared to using 100% labeled samples.

8.2 Future Work

In this thesis, it was successfully demonstrated that the co-training method could be extended using several machine learning techniques like multi-label classification, matrix factorization, recommendation and finally deep learning. There are several interesting future lines of research. Let us present some of them:

- Feature engineering for detailed feedback - Matrix factorization algorithms can take advantage of row and column features if they are available. It has also been shown that good results can also be achieved without matrix factorization, but further improvements can still be made. To improve the effectiveness of the methods, feature extraction methods based on static code analysis should be used to train only the most dominant features and not the entire labeled dataset. This approach would require less computational power and might make it easier for this model to be trained and predict more accurately.

- Recommendations using hierarchies: The rows or columns of the input matrix often form a hierarchy. It is possible for recommender systems to generate hierarchies based on factors for instance movie sequels, editions, or, for users, users' addresses. Software engineering applications almost naturally generate hierarchy structures. Similar analogies can be made with recommender systems. They are -

  - The description of a product is part of its identification; this is part of a class, which is in turn organized into topics, demographics or areas.
  - Products are members of a set, which belong to a department, which belongs to a company.

  In addition to increasing a model's predictive performance as a result of these and other hierarchies, they can also let it degrade in a more gradual way by recommending a movie from the same genre if the most likely one is not suggested. Factor models

do not currently use this information. Research could then investigate the transfer, adaptation, and augmentation of these results to factor models.

- Factor models with context-aware biases: Recently, a factor model was proposed for Recommender Systems that considers the temporal effects of recommendation systems by varying the row and column biases with time Lin and Chen (2019). The rationale behind recommender systems is that users' preferences change over time, and the overall preference changes over time due to successful product placement in national TV shows. In a broader perspective, this recent development hinted at the possibility of context-aware factor models. The proposal can be incorporated into our thesis framework by having different column and row biases for different times. The row and column biases in the example above are connected by reference to time. It is possible to extend this concept to include matrices. As an example, if you were recommending a movie, the most significant of these features would be the time at which the rating is given or expected to be given. A second source of context for the Recommender System could be the intended audience of the movie, since recommendations for a family might differ depending on whether a movie is intended just for the parents or for the whole family. By using the user's features as context, such a system allows tailoring the suggestions to the individual. Accordingly, the system would shift from generic predictions for instance "In other instances, people also did A." to more personalized ones for instance "In other instances, people like you also did A." This raises questions in the application domain, for instance how to define a similarity metric within learners and under what circumstances a personalization facilitates or hinders learning.

- Domain independent self-learning online system - It is clear that our model includes some assumptions, for instance the fact that experts provided us with annotated lists of human rights violations before the model was developed. It would be more beneficial to design an online system that would allow domain experts to upload their

annotations and an unlabeled dataset. By learning from the annotations, the model would understand what domain the annotation pertains to, and proceed to classify the unlabeled dataset based on the annotations. This means that the model's scope is not limited to only human rights, but rather should be independent of any domain, with the initial labeled set being derived primarily from expert annotation.

List of References

Abney, S.: 2002, Bootstrapping, *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 360–367.
URL: *https://doi.org/10.3115/1073083.1073143*

Adams, P. A., Adrian, T., Boyarchenko, N. and Giannone, D.: 2021, Forecasting macroeconomic risks, *International Journal of Forecasting* **37**(3), 1173–1191.

Adomavicius, G. and Tuzhilin, A.: 2005, Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions, *IEEE transactions on knowledge and data engineering* **17**(6), 734–749.

Ahmad, T. and Doja, M. N.: 2013, Opinion mining using frequent pattern growth method from unstructured text, *2013 International Symposium on Computational and Business Intelligence*, IEEE, pp. 92–95.

Ahmed, F., Iqbal, M. F. and Rafiq, A.: 2019, On cone optimization approaches for semi-supervised support vector machines (s 3 vm), *2019 2nd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, IEEE, pp. 1–6.

Akas, M. F., Zaman, A. and Khan, A.: 2020, Combined item sets generation using modified apriori algorithm, *Proceedings of the International Conference on Computing Advancements*, pp. 1–3.

Albawi, S., Mohammed, T. A. and Al-Zawi, S.: 2017, Understanding of a convolutional neural network, *2017 International Conference on Engineering and Technology (ICET)*, Ieee, pp. 1–6.

Alloghani, M., Al-Jumeily, D., Mustafina, J., Hussain, A. and Aljaaf, A. J.: 2020, A systematic review on supervised and unsupervised machine learning algorithms for data science, *Supervised and unsupervised learning for data science* pp. 3–21.

Alsharif, M. H., Kelechi, A. H., Yahya, K. and Chaudhry, S. A.: 2020, Machine learning algorithms for smart data analysis in internet of things environment: taxonomies and research trends, *Symmetry* **12**(1), 88.

Alvarsson, J., McShane, S. A., Norinder, U. and Spjuth, O.: 2021, Predicting with confidence: using conformal prediction in drug discovery, *Journal of Pharmaceutical Sciences* **110**(1), 42–49.

Aly, M.: 2005, Survey on multiclass classification methods, *Neural Netw* **19**, 1–9.

Ariyo, A. A., Adewumi, A. O. and Ayo, C. K.: 2014, Stock price prediction using the arima model, *2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation*, IEEE, pp. 106–112.

Arora, S., Athavale, V. A., Maggu, H. and Agarwal, A.: 2021, Artificial intelligence and virtual assistantworking model, *Mobile Radio Communications and 5G Networks*, Springer, pp. 163–171.

Awasthi, I., Gupta, K., Bhogal, P. S., Anand, S. S. and Soni, P. K.: 2021, Natural language processing (nlp) based text summarization-a survey, *2021 6th International Conference on Inventive Computation Technologies (ICICT)*, IEEE, pp. 1310–1317.

Baeza-Yates, R., Ribeiro-Neto, B. et al.: 1999, *Modern information retrieval*, Vol. 463, ACM press New York.

Balogh, V., Berend, G., Diochnos, D. I. and Turán, G.: 2020, Understanding the semantic content of sparse word embeddings using a commonsense knowledge base, *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34, pp. 7399–7406.

Bell, P., Fainberg, J., Klejch, O., Li, J., Renals, S. and Swietojanski, P.: 2020, Adaptation algorithms for neural network-based speech recognition: An overview, *IEEE Open Journal of Signal Processing* **2**, 33–66.

Ben-David, A.: 2007, A lot of randomness is hiding in accuracy, *Engineering Applications of Artificial Intelligence* **20**(7), 875–885.

Bengio, Y.: 2012, Deep learning of representations for unsupervised and transfer learning, *Proceedings of ICML workshop on unsupervised and transfer learning*, JMLR Workshop and Conference Proceedings, pp. 17–36.

Bengio, Y., Courville, A. and Vincent, P.: 2013, Representation learning: A review and new perspectives, *IEEE transactions on pattern analysis and machine intelligence* **35**(8), 1798–1828.

Blum, A. and Mitchell, T.: 1998, Combining labeled and unlabeled data with co-training, *Proceedings of the eleventh annual conference on Computational learning theory*, ACM, pp. 92–100.

Bolognesi, M. and Aina, L.: 2019, Similarity is closeness: Using distributional semantic spaces to model similarity in visual and linguistic metaphors, *Corpus Linguistics and Linguistic Theory* **15**(1), 101–137.
URL: *https://doi.org/10.1515/cllt-2016-0061*

Boukerche, A. and Wang, J.: 2020, Machine learning-based traffic prediction models for intelligent transportation systems, *Computer Networks* **181**, 107530.

Boutell, M., Shen, X., Luo, J. and Brown, C.: 2003, Multi-label semantic scene classification, *Technical report*, Citeseer.

Business, T. and Centre, L. G. D. R.: n.d., The business and local government data research centre, `TheBusinessandLocalGovernmentDataResearchCentre`.

Chan, J., Koprinska, I. and Poon, J.: 2004, Co-training with a single natural feature set applied to email classification, *IEEE/WIC/ACM International Conference on Web Intelligence (WI'04)*, IEEE, pp. 586–589.

Chang, C.-C. and Lin, C.-J.: 2011, Libsvm: a library for support vector machines, *ACM transactions on intelligent systems and technology (TIST)* **2**(3), 1–27.

Chen, C., Li, D., Lv, Q., Yan, J., Chu, S. M. and Shang, L.: 2016, Mpma: Mixture probabilistic matrix approximation for collaborative filtering., *IJCAI*, pp. 1382–1388.

Chen, M., Xu, Z., Weinberger, K. and Sha, F.: 2012, Marginalized denoising autoencoders for domain adaptation, *arXiv preprint arXiv:1206.4683* .

Chong, Y., Ding, Y., Yan, Q. and Pan, S.: 2020, Graph-based semi-supervised learning: A review, *Neurocomputing* **408**, 216–230.

Chowdhury, N. and Saha, D.: 2005, Unsupervised text classification using kohonens self organizing network, *International Conference on Intelligent Text Processing and Computational Linguistics*, Springer, pp. 715–718.

Cinar, A. C.: 2020, Training feed-forward multi-layer perceptron artificial neural networks with a tree-seed algorithm, *Arabian Journal for Science and Engineering* **45**(12), 10915–10938.

Clare, A. and King, R. D.: 2001, Knowledge discovery in multi-label phenotype data, *European Conference on Principles of Data Mining and Knowledge Discovery*, Springer, pp. 42–53.

Da Costa, A. F., Manzato, M. G. and Campello, R. J.: 2019, Boosting collaborative filtering with an ensemble of co-trained recommenders, *Expert Systems with Applications* **115**, 427–441.

Dalal, M. K. and Zaveri, M. A.: 2011, Automatic text classification: a technical review, *International Journal of Computer Applications* **28**(2), 37–40.

Davis, J. and Goadrich, M.: 2006, The relationship between precision-recall and roc curves, *Proceedings of the 23rd international conference on Machine learning*, ACM, pp. 233–240.

Dempster, A. P., Laird, N. M. and Rubin, D. B.: 1977, Maximum likelihood from incomplete data via the em algorithm, *Journal of the Royal Statistical Society: Series B (Methodological)* **39**(1), 1–22.

Dey, A.: 2016, Machine learning algorithms: a review, *International Journal of Computer Science and Information Technologies* **7**(3), 1174–1179.

Dharmadhikari, S., Ingle, M. and Kulkarni, P.: 2012, Analysis of semi supervised learning methods towards multi label text classification, *International Journal of Computer Applications* **42**(16), 15–20.

Dietterich, T. G.: 1998, Approximate statistical tests for comparing supervised classification learning algorithms, *Neural computation* **10**(7), 1895–1923.

Edunov, S., Ott, M., Auli, M. and Grangier, D.: 2018, Understanding back-translation at scale, *arXiv preprint arXiv:1808.09381* .

Elisseeff, A. and Weston, J.: 2002, A kernel method for multi-labelled classification, *Advances in neural information processing systems*, pp. 681–687.

Elreedy, D. and Atiya, A. F.: 2019, A comprehensive analysis of synthetic minority oversampling technique (smote) for handling class imbalance, *Information Sciences* **505**, 32–64.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J. and Zisserman, A.: 2010, The pascal visual object classes (voc) challenge, *International journal of computer vision* **88**(2), 303–338.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R. and Lin, C.-J.: 2008, Liblinear: A library for large linear classification, *the Journal of machine Learning research* **9**, 1871–1874.

Fang, J.: 2020, A critical review of five machine learning-based algorithms for predicting protein stability changes upon mutation, *Briefings in bioinformatics* **21**(4), 1285–1292.

Fasanmade, A., He, Y., Al-Bayatti, A. H., Morden, J. N., Aliyu, S. O., Alfakeeh, A. S. and Alsayed, A. O.: 2020, A fuzzy-logic approach to dynamic bayesian severity level classification of driver distraction using image recognition, *IEEE Access* **8**, 95197–95207.

Fleder, D. M. and Hosanagar, K.: 2007, Recommender systems and their impact on sales diversity, *Proceedings of the 8th ACM conference on Electronic commerce*, ACM, pp. 192–199.

Freund, Y. and Schapire, R. E.: 1997, A decision-theoretic generalization of on-line learning and an application to boosting, *Journal of computer and system sciences* **55**(1), 119–139.

Freytag, A., Rodner, E., Bodesheim, P. and Denzler, J.: 2013, Labeling examples that matter: Relevance-based active learning with gaussian processes, *German Conference on Pattern Recognition*, Springer, pp. 282–291.

Gambus, P. and Shafer, S. L.: 2018, Artificial intelligence for everyone, *Anesthesiology* **128**(3), 431–433.

Gokhale, R. and Fasli, M.: 2017a, Deploying a co-training algorithm to classify human-rights abuses, *2017 International Conference on the Frontiers and Advances in Data Science (FADS)*, IEEE, pp. 108–113.

Gokhale, R. and Fasli, M.: 2017b, Deploying a co-training algorithm to classify human-rights abuses, *2017 International Conference on the Frontiers and Advances in Data Science (FADS)*, pp. 108–113.

Gokhale, R. and Fasli, M.: 2018, Matrix factorization for co-training algorithm to classify human rights abuses, *2018 IEEE International Conference on Big Data (Big Data)*, IEEE, pp. 2170–2179.

Goldman, S. and Zhou, Y.: 2000, Enhancing supervised learning with unlabeled data, *ICML*, Citeseer, pp. 327–334.

Gonçalves, T. and Quaresma, P.: 2003, A preliminary approach to the multilabel classification problem of portuguese juridical documents, *Portuguese Conference on Artificial Intelligence*, Springer, pp. 435–444.

Haldorai, A. and Ramu, A.: 2021, Canonical correlation analysis based hyper basis feedforward neural network classification for urban sustainability, *Neural Processing Letters* **53**(4), 2385–2401.

Han, F., Yao, J., Zhu, H. and Wang, C.: 2020, Underwater image processing and object detection based on deep cnn method, *Journal of Sensors* **2020**.

He, H., Han, D. and Dezert, J.: 2020, Disagreement based semi-supervised learning approaches with belief functions, *Knowledge-Based Systems* **193**, 105426.

Herlocker, J. L., Konstan, J. A., Terveen, L. G. and Riedl, J. T.: 2004, Evaluating collaborative filtering recommender systems, **22**(1), 5–53.

Hewage, P., Trovati, M., Pereira, E. and Behera, A.: 2021, Deep learning-based effective fine-grained weather forecasting model, *Pattern Analysis and Applications* **24**(1), 343–366.

Hewamalage, H., Bergmeir, C. and Bandara, K.: 2021, Recurrent neural networks for time series forecasting: Current status and future directions, *International Journal of Forecasting* **37**(1), 388–427.

Hordri, N. F., Yuhaniz, S. S., Azmi, N. F. M. and Shamsuddin, S. M.: 2018, Handling class imbalance in credit card fraud using resampling methods, *Int. J. Adv. Comput. Sci. Appl* **9**(11), 390–396.

HRDAG: n.d., Hrdag human rights data analysis group, `https://hrdag.org/`. Accessed June 14, 2020.

Huang, F., Zhang, J., Zhou, C., Wang, Y., Huang, J. and Zhu, L.: 2020, A deep learning algorithm using a fully connected sparse autoencoder neural network for landslide susceptibility prediction, *Landslides* **17**(1), 217–229.

Huang, J., Lu, J. and Ling, C. X.: 2003, Comparing naive bayes, decision trees, and svm with auc and accuracy, *Third IEEE International Conference on Data Mining*, IEEE, pp. 553–556.

Huang, X.: 2013, Andrew Ng: Deep Learning, Self-Taught Learning and Unsupervised Feature Learning. Andrew Ng: Deep Learning 2013.
URL: *https://www.youtube.com/watch?v=n1ViNeWhC24*

Huang, Y. and Li, L.: 2011, Naive bayes classification algorithm based on small sample set, *2011 IEEE International conference on cloud computing and intelligence systems*, IEEE, pp. 34–39.

HURIDOCS: n.d., Huridocs, `https://www.huridocs.org/2018/05/`

`starting-at-the-source-introducing-uwazi-reveal/`. Accessed September 4, 2019.

Imai, C., Armstrong, B., Chalabi, Z., Mangtani, P. and Hashizume, M.: 2015, Time series regression model for infectious disease and weather, *Environmental research* **142**, 319–327.

Jaakkola, T. S. and Haussler, D.: 1999, Probabilistic kernel regression models, *Seventh International Workshop on Artificial Intelligence and Statistics*, PMLR.

Jani, K., Chaudhuri, M., Patel, H. and Shah, M.: 2020, Machine learning in films: an approach towards automation in film censoring, *Journal of Data, Information and Management* **2**(1), 55–64.

Joachims, T.: 1998, Text categorization with support vector machines: Learning with many relevant features, *European conference on machine learning*, Springer, pp. 137–142.

Joachims, T. et al.: 1999, Transductive inference for text classification using support vector machines, *Icml*, Vol. 99, pp. 200–209.

Jolliffe, I. T. and Cadima, J.: 2016, Principal component analysis: a review and recent developments, *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* **374**(2065), 20150202.

Karalic, A. and Pirnat, V.: 1991, Significance level based multiple tree classification, *Informatica* **15**(5), 12.

Khalid, H., Hussain, M., Al Ghamdi, M. A., Khalid, T., Khalid, K., Khan, M. A., Fatima, K., Masood, K., Almotiri, S. H., Farooq, M. S. et al.: 2020, A comparative systematic literature review on knee bone reports from mri, x-rays and ct scans using deep learning and machine learning methodologies, *Diagnostics* **10**(8), 518.

Khan, A., Baharudin, B., Lee, L. H. and Khan, K.: 2010, A review of machine learning algorithms for text-documents classification, *Journal of advances in information technology* **1**(1), 4–20.

Khan, A., Gul, M. A., Uddin, M. I., Ali Shah, S. A., Ahmad, S., Firdausi, A., Dzulqarnain, M. and Zaindin, M.: 2020, Summarizing online movie reviews: a machine learning approach to big data analytics, *Scientific Programming* **2020**.

Kharde, V., Sonawane, P. et al.: 2016, Sentiment analysis of twitter data: a survey of techniques, *arXiv preprint arXiv:1601.06971* .

Kihlman, R. and Fasli, M.: 2019, Augmenting co-training with recommendations to classify human rights violations, *2019 IEEE International Conference on Big Data (Big Data)*, IEEE, pp. 2299–2308.

Kihlman, R. and Fasli, M.: 2021, Classifying human rights violations using deep multi-label co-training, *2021 IEEE International Conference on Big Data (Big Data)*, IEEE, pp. 4887–4895.

Kiritchenko, S. and Matwin, S.: 2001, Email classification with co-training, *Proceedings of the 2001 conference of the Centre for Advanced Studies on Collaborative research*, Citeseer, p. 8.

Konstan, J. A. and Riedl, J.: 2012, Recommender systems: from algorithms to user experience, *User modeling and user-adapted interaction* **22**(1), 101–123.

Koren, Y., Bell, R. and Volinsky, C.: 2009, Matrix factorization techniques for recommender systems, *Computer* **42**(8), 30–37.

Kosko, B.: 1988, Hidden patterns in combined and adaptive knowledge networks, *International Journal of Approximate Reasoning* **2**(4), 377–393.

Kotsiantis, S. B.: 2013, Decision trees: a recent overview, *Artificial Intelligence Review* **39**(4), 261–283.

Kowsari, K., Brown, D. E., Heidarysafa, M., Meimandi, K. J., Gerber, M. S. and Barnes, L. E.: 2017, Hdltex: Hierarchical deep learning for text classification, *2017 16th IEEE international conference on machine learning and applications (ICMLA)*, IEEE, pp. 364–371.

Kratsch, W., Manderscheid, J., Röglinger, M. and Seyfried, J.: 2020, Machine learning in business process monitoring: a comparison of deep learning and classical approaches used for outcome prediction, *Business & Information Systems Engineering* pp. 1–16.

Kumari, R. and Srivastava, S. K.: 2017, Machine learning: A review on binary classification, *International Journal of Computer Applications* **160**(7).

Laishram, A., Padmanabhan, V. and Lal, R. P.: 2018, Analysis of similarity measures in user-item subgroup based collaborative filtering via genetic algorithm, *International Journal of Information Technology* **10**(4), 523–527.

Lee, J.-S., Jun, C.-H., Lee, J. and Kim, S.: 2005, Classification-based collaborative filtering using market basket data, *Expert systems with applications* **29**(3), 700–704.

Li, Y., Guo, H., Zhang, Q., Gu, M. and Yang, J.: 2018, Imbalanced text sentiment classification using universal and domain-specific knowledge, *Knowledge-Based Systems* **160**, 1–15.

Liang, H., Sun, X., Sun, Y. and Gao, Y.: 2017, Text feature extraction based on deep learning: a review, *EURASIP journal on wireless communications and networking* **2017**(1), 1–12.

Lin, Z. and Chen, H.: 2019, A probabilistic model for collaborative filtering, *Proceedings of the 9th International Conference on Web Intelligence, Mining and Semantics*, pp. 1–8.

Liu, T., Liu, S., Chen, Z. and Ma, W.-Y.: 2003, An evaluation on feature selection for text clustering, *Proceedings of the 20th international conference on machine learning (ICML-03)*, pp. 488–495.

Liu, Y., Chen, H., Chen, Y., Yin, W. and Shen, C.: 2021, Generic perceptual loss for modeling structured output dependencies, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5424–5432.

Livieris, I. E., Pintelas, E. and Pintelas, P.: 2020, A cnn–lstm model for gold price time-series forecasting, *Neural computing and applications* **32**(23), 17351–17360.

Luo, X. and Zincir-Heywood, A. N.: 2005, Evaluation of two systems on multi-class multi-label document classification, *International Symposium on Methodologies for Intelligent Systems*, Springer, pp. 161–169.

Ma, F., Meng, D., Xie, Q., Li, Z. and Dong, X.: 2017, Self-paced co-training, *in* D. Precup and Y. W. Teh (eds), *Proceedings of the 34th International Conference on Machine Learning*, Vol. 70 of *Proceedings of Machine Learning Research*, PMLR, pp. 2275–2284. **URL:** *https://proceedings.mlr.press/v70/ma17b.html*

Mangalindan, J.: 2012, Amazons recommendation secret, *CNN Money http://tech. fortune. cnn. com/2012/07/30/amazon-5* .

Matloff, N.: 2017, *Statistical regression and classification: from linear models to machine learning*, CRC Press.

McAuley, J., Leskovec, J. and Jurafsky, D.: 2012a, Learning attitudes and attributes from multi-aspect reviews, *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, IEEE, pp. 1020–1025.

McAuley, J., Leskovec, J. and Jurafsky, D.: 2012b, Learning attitudes and attributes from multi-aspect reviews, *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, IEEE, pp. 1020–1025.

McDonald, R., Crammer, K. and Pereira, F.: 2005, Flexible text segmentation with structured multilabel classification, *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, pp. 987–994.

Meyer, W. H.: 1996, Human rights and mncs: Theory versus quantitative analysis, *Hum. Rts. Q.* **18**, 368.

Meyer, W. H.: n.d., Human rights and mncs: Theory versus quantitative analysis(1996), *Human Rights Quarterly* **18**, 368.

Motwani, M., Dey, D., Berman, D. S., Germano, G., Achenbach, S., Al-Mallah, M. H., Andreini, D., Budoff, M. J., Cademartiri, F., Callister, T. Q. et al.: 2017, Machine learning for prediction of all-cause mortality in patients with suspected coronary artery disease: a 5-year multicentre prospective registry analysis, *European heart journal* **38**(7), 500–507.

Mu, R.: 2018, A survey of recommender systems based on deep learning, *Ieee Access* **6**, 69009–69022.

Müller, M., Salathé, M. and Kummervold, P. E.: 2020, Covid-twitter-bert: A natural language processing model to analyse covid-19 content on twitter, *arXiv preprint arXiv:2005.07503* .

Murty, M. R., Murthy, J. and PVGD, P. R.: 2011, Text document classification based-on least square support vector machines with singular value decomposition, *International Journal of Computer Applications* **27**(7), 21–26.

Nallaperuma, D., Nawaratne, R., Bandaragoda, T., Adikari, A., Nguyen, S., Kempitiya, T., De Silva, D., Alahakoon, D. and Pothuhera, D.: 2019, Online incremental machine learning platform for big data-driven smart traffic management, *IEEE Transactions on Intelligent Transportation Systems* **20**(12), 4679–4690.

Nguyen, G., Dlugolinsky, S., Bobák, M., Tran, V., García, Á. L., Heredia, I., Malík, P. and Hluchỳ, L.: 2019, Machine learning and deep learning frameworks and libraries for large-scale data mining: a survey, *Artificial Intelligence Review* **52**(1), 77–124.

Nigam, K. and Ghani, R.: 2000, Analyzing the effectiveness and applicability of co-training, *Proceedings of the ninth international conference on Information and knowledge management*, pp. 86–93.

Nigam, K., McCallum, A. K., Thrun, S. and Mitchell, T.: 2000, Text classification from labeled and unlabeled documents using em, *Machine learning* **39**(2), 103–134.

Nilashi, M., Esfahani, M. D., Roudbaraki, M. Z., Ramayah, T. and Ibrahim, O.: 2016, A multi-criteria collaborative filtering recommender system using clustering and regression techniques, *Journal of Soft Computing and Decision Support Systems* **3**(5), 24–30.

Niu, M., Li, Y., Wang, C. and Han, K.: 2018, Rfamyloid: A web server for predicting amyloid proteins, *International journal of molecular sciences* **19**.

Ogheneovo, E. and Japheth, R.: 2016, Application of vector space model to query ranking and information retrieval, *International Journal of Advanced Research in Computer Science and Software Engineering* **6**(5).

Pant, P., Sabitha, A. S., Choudhury, T. and Dhingra, P.: 2019, Multi-label classification trending challenges and approaches, *Emerging Trends in Expert Applications and Security*, Springer, pp. 433–444.

Park, D. S., Zhang, Y., Jia, Y., Han, W., Chiu, C.-C., Li, B., Wu, Y. and Le, Q. V.: 2020, Improved noisy student training for automatic speech recognition, *arXiv preprint arXiv:2005.09629* .

Patel, J., Shah, S., Thakkar, P. and Kotecha, K.: 2015, Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques, *Expert systems with applications* **42**(1), 259–268.

Pierce, D. and Cardie, C.: 2001, Limitations of co-training for natural language learning from large datasets, *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*.

Polatidis, N. and Georgiadis, C. K.: 2016, A multi-level collaborative filtering method that improves recommendations, *Expert Systems with Applications* **48**, 100–110.

Portugal, I., Alencar, P. and Cowan, D.: 2018, The use of machine learning algorithms in recommender systems: A systematic review, *Expert Systems with Applications* **97**, 205–227.

Project, C. H. R. D.: n.d., Ciri human rights data project, `www.humanrightsdata.com`. Accessed June 14, 2020.

Qiao, S., Shen, W., Zhang, Z., Wang, B. and Yuille, A.: 2018, Deep co-training for semi-supervised image recognition, *Proceedings of the european conference on computer vision (eccv)*, pp. 135–152.

Ramezani, M., Tab, F. A., Abdollahpouri, A. and Mohammad, M. A.: 2021, A new generalized collaborative filtering approach on sparse data by extracting high confidence relations between users, *Information Sciences* **570**, 323–341.

Raskutti, B., Ferra, H. and Kowalczyk, A.: 2002, Combining clustering and co-training to enhance text classification using unlabelled data, *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 620–625.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P. and Riedl, J.: 1994, Grouplens: an open architecture for collaborative filtering of netnews, *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, ACM, pp. 175–186.

Resnick, P. and Varian, H. R.: 1997, Recommender systems, *Communications of the ACM* **40**(3), 56–58.

Rivas, A., Chamoso, P., González-Briones, A., Pavón, J. and Corchado, J. M.: 2020, Social network recommender system, a neural network approach, *International Conference on Intelligent Data Engineering and Automated Learning*, Springer, pp. 213–222.

Rodrigues, R.: 2020, Legal and human rights issues of ai: gaps, challenges and vulnerabilities, *Journal of Responsible Technology* **4**, 100005.

Rosenberg, C., Hebert, M. and Schneiderman, H.: 2005, Semi-supervised self-training of object detection models.

Sahu, A. K. and Dwivedi, P.: 2019, User profile as a bridge in cross-domain recommender systems for sparsity reduction, *Applied Intelligence* **49**(7), 2461–2481.

Sankaran, P., Sunoj, S. and Nair, N. U.: 2016, Kullback–leibler divergence: A quantile approach, *Statistics & Probability Letters* **111**, 72–79.

Saravanan, R. and Sujatha, P.: 2018, A state of art techniques on machine learning algorithms: a perspective of supervised learning approaches in data classification, *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*, IEEE, pp. 945–949.

Sarddar, D., Dey, R. K., Bose, R. and Roy, S.: 2020, Topic modeling as a tool to gauge political sentiments from twitter feeds, *International Journal of Natural Computing Research (IJNCR)* **9**(2), 14–35.

Sarwar, B., Karypis, G., Konstan, J. and Riedl, J.: 2002, Incremental singular value decomposition algorithms for highly scalable recommender systems, *Fifth International Conference on Computer and Information Science*, Citeseer, pp. 27–28.

Schafer, J. B., Konstan, J. A. and Riedl, J.: 2001, E-commerce recommendation applications, *Data mining and knowledge discovery* **5**(1-2), 115–153.

Schein, A. I., Popescul, A., Ungar, L. H. and Pennock, D. M.: 2002, Methods and metrics for cold-start recommendations, *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 253–260.

Sechidis, K., Tsoumakas, G. and Vlahavas, I.: 2011, On the stratification of multi-label data, *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Springer, pp. 145–158.

Seger, C.: 2018, An investigation of categorical variable encoding techniques in machine learning: binary versus one-hot and feature hashing.

Shen, X., Boutell, M., Luo, J. and Brown, C.: 2003, Multilabel machine learning and its application to semantic scene classification, *Storage and Retrieval Methods and Applications for Multimedia 2004*, Vol. 5307, International Society for Optics and Photonics, pp. 188–200.

Singh, S. K., Dwivedi, D. and Kumar, R.: 2020, Data mining: Dirty data and data cleaning, *Data Mining: Dirty Data and Data Cleaning (May 26, 2020)* .

Singh, S. P., Kumar, A., Mangal, A. and Singhal, S.: 2016, Bilingual automatic text summarization using unsupervised deep learning, *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, IEEE, pp. 1195–1200.

Singh, V., Kumar, B. and Patnaik, T.: 2013, Feature extraction techniques for handwritten text in various scripts: a survey, *International Journal of Soft Computing and Engineering (IJSCE)* **3**(1), 238–241.

Sivakumar, S. and Rajalakshmi, R.: 2019, Comparative evaluation of various feature weighting methods on movie reviews, *Computational intelligence in data mining*, Springer, pp. 721–730.

Stahlberg, F.: 2020, Neural machine translation: A review, *Journal of Artificial Intelligence Research* **69**, 343–418.

Stoyanova, M. et al.: 2020, Good practices and recommendations for success in construction digitalization, *TEM Journal* **9**(1), 42–47.

Sun, S.-B., Zhang, Z.-H., Dong, X.-L., Zhang, H.-R., Li, T.-J., Zhang, L. and Min, F.: 2017, Integrating triangle and jaccard similarities for recommendation, *PloS one* **12**(8), e0183570.

Suthaharan, S.: 2014, Big data classification: Problems and challenges in network intrusion prediction with machine learning, *ACM SIGMETRICS Performance Evaluation Review* **41**(4), 70–73.

Tahmassebi, A., Gandomi, A. H., McCann, I., Schulte, M. H., Goudriaan, A. E. and Meyer-Baese, A.: 2018, Deep learning in medical imaging: fmri big data analysis via convolutional neural networks, *Proceedings of the Practice and Experience on Advanced Research Computing*, ACM press New York, pp. 1–4.

Tharwat, A.: 2020, Classification assessment methods, *Applied Computing and Informatics* .

Thinyane, H. and Sassetti, F.: 2020, Towards a human rights-based approach to ai: Case study of apprise, *International Development Informatics Association Conference*, Springer, pp. 33–47.

Tsai, C.-W., Lai, C.-F., Chao, H.-C. and Vasilakos, A. V.: 2015, Big data analytics: a survey, *Journal of Big data* **2**(1), 1–32.

Tsoumakas, G. and Katakis, I.: 2007, Multi-label classification: An overview, *International Journal of Data Warehousing and Mining (IJDWM)* **3**(3), 1–13.

Tsoumakas, G., Katakis, I. and Vlahavas, I.: 2009, Mining multi-label data, *Data mining and knowledge discovery handbook*, Springer, pp. 667–685.

Uslan, V. and Seker, H.: 2016, Quantitative prediction of peptide binding affinity by using hybrid fuzzy support vector regression, *Applied Soft Computing* **43**, 210–221.

Van Engelen, J. E. and Hoos, H. H.: 2020, A survey on semi-supervised learning, *Machine Learning* **109**(2), 373–440.

Van Meteren, R. and Van Someren, M.: 2000, Using content-based filtering for recommendation, *Proceedings of the Machine Learning in the New Information Age: MLnet/ECML2000 Workshop*, pp. 47–56.

Vijaymeena, M. and Kavitha, K.: 2016, A survey on similarity measures in text mining, *Machine Learning and Applications: An International Journal* **3**(2), 19–28.

Wang, H. and Raj, B.: 2017, On the origin of deep learning, *arXiv preprint arXiv:1702.07800* .

Wang, W., Lu, D., Zhou, X., Zhang, B. and Mu, J.: 2013, Statistical wavelet-based anomaly detection in big data with compressive sensing, *EURASIP Journal on Wireless Communications and Networking* **2013**(1), 1–6.

Wang, W. and Zhou, Z.-H.: 2007, Analyzing co-training style algorithms, *European conference on machine learning*, Springer, pp. 454–465.

Wang, W. and Zhou, Z.-H.: 2017, Theoretical foundation of co-training and disagreement-based algorithms, *arXiv preprint arXiv:1708.04403* .

Wei, L. and Keogh, E.: 2006, Semi-supervised time series classification, *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 748–753.

Wei, Q., Franklin, A., Cohen, T. and Xu, H.: 2018, Clinical text annotation–what factors are associated with the cost of time?, *AMIA Annual Symposium Proceedings*, Vol. 2018, American Medical Informatics Association, p. 1552.

WITNESS: n.d., Witness, `https://www.witness.org/`. Accessed June 14, 2020.

Wong, T.-T.: 2015, Performance evaluation of classification algorithms by k-fold and leave-one-out cross validation, *Pattern Recognition* **48**(9), 2839–2846.

Wu, G. and Zhu, J.: 2020, Multi-label classification: do hamming loss and subset accuracy really conflict with each other?, *arXiv preprint arXiv:2011.07805* .

Wu, J.: 2012, Cluster analysis and k-means clustering: an introduction, *Advances in K-means Clustering*, Springer, pp. 1–16.

Wu, Y. and Ester, M.: 2015, Flame: A probabilistic model combining aspect based opinion mining and collaborative filtering, *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pp. 199–208.

Xu, C., Tao, D. and Xu, C.: 2013, A survey on multi-view learning, *arXiv preprint arXiv:1304.5634* .

Yao, X.: 1993, A review of evolutionary artificial neural networks, *International journal of intelligent systems* **8**(4), 539–567.

Yu, K., Zhu, S., Lafferty, J. and Gong, Y.: 2009, Fast nonparametric matrix factorization for large-scale collaborative filtering, *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, ACM, pp. 211–218.

Zhang, J., Lin, Y., Lin, M. and Liu, J.: 2016, An effective collaborative filtering algorithm based on user preference clustering, *Applied Intelligence* **45**(2), 230–240.

Zhang, M.-L., Li, Y.-K., Yang, H. and Liu, X.-Y.: 2020, Towards class-imbalance aware multi-label learning, *IEEE Transactions on Cybernetics* .

Zhang, M.-L. and Zhou, Z.-H.: 2005, A k-nearest neighbor based algorithm for multi-label classification, *Granular Computing, 2005 IEEE International Conference on*, Vol. 2, IEEE, pp. 718–721.

Zhang, M.-L. and Zhou, Z.-H.: 2011, Cotrade: Confident co-training with data editing, *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* **41**(6), 1612–1626.

Zhang, Q., Lu, J. and Jin, Y.: 2021, Artificial intelligence in recommender systems, *Complex & Intelligent Systems* **7**(1), 439–457.

Zheng, Y., Capra, L., Wolfson, O. and Yang, H.: 2014, Urban computing: concepts, methodologies, and applications, *ACM Transactions on Intelligent Systems and Technology (TIST)* **5**(3), 1–55.

Zhou, G., Sohn, K. and Lee, H.: 2012, Online incremental feature learning with denoising autoencoders, *Artificial intelligence and statistics*, PMLR, pp. 1453–1461.

Zhou, Y. and Goldman, S.: 2004, Democratic co-learning, *16th IEEE International Conference on Tools with Artificial Intelligence*, IEEE, pp. 594–602.

Zhou, Y., Wilkinson, D., Schreiber, R. and Pan, R.: 2008, Large-scale parallel collaborative filtering for the netflix prize, *International Conference on Algorithmic Applications in Management*, Springer, pp. 337–348.

Zhou, Z.-H. and Li, M.: 2005, Tri-training: Exploiting unlabeled data using three classifiers, *IEEE Transactions on knowledge and Data Engineering* **17**(11), 1529–1541.

Zhou, Z.-H. and Li, M.: 2007, Semisupervised regression with cotraining-style algorithms, *IEEE Transactions on Knowledge and Data Engineering* **19**(11), 1479–1493.

Zhou, Z.-H. and Li, M.: 2010, Semi-supervised learning by disagreement, *Knowledge and Information Systems* **24**(3), 415–439.

Zhu, S., Yu, K., Chi, Y. and Gong, Y.: 2007, Combining content and link for classification using matrix factorization, *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, ACM, pp. 487–494.

Zhu, X. and Goldberg, A. B.: 2009, Introduction to semi-supervised learning, *Synthesis lectures on artificial intelligence and machine learning* **3**(1), 1–130.