PATTERN RECOGNITION AND QUANTIFYING ASSOCIATIONS WITHIN ENTITIES OF DATA DRIVEN SYSTEMS FOR IMPROVING MODEL INTERPRETABILITY

A Dissertation
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Arighna Roy

In Partial Fulfillment of the Requirements
for the Degree of
DOCTOR OF PHILOSOPHY

Major Department:
Computer Science

April 2022

Fargo, North Dakota

NORTH DAKOTA STATE UNIVERSITY

Graduate School

Title

PATTERN RECOGNITION AND QUANTIFYING ASSOCIATIONS WITHIN ENTITIES OF DATA DRIVEN SYSTEMS FOR IMPROVING MODEL INTERPRETABILITY

By

Arighna Roy

The supervisory committee certifies that this dissertation complies with North Dakota State University's regulations and meets the accepted standards for the degree of

DOCTOR OF PHILOSOPHY

SUPERVISORY COMMITTEE:

Dr. Simone Ludwig
Chair

Dr. Saeed Salem

Dr. Anne Denton

Dr. Rhonda Magel

Approved:

April 20, 2022
Date

Dr. Simone Ludwig
Department Chair

ABSTRACT

Discovering associations among entities of a system plays an important role in data science. The majority of the data science related problems have become heavily dependent on Machine Learning (ML) since the rise of computation power. However, the majority of the machine learning approaches rely on improving the performance of the algorithm by optimizing an objective function, at the cost of compromising the interpretability of the models. A new branch of machine learning focuses on model interpretability by explaining the models in various ways. The foundation of model interpretability is built on extracting patterns from the behavior of the models and the related entities. Gradually, Machine learning has spread its wing to almost every industry. This dissertation focuses on the data science application to three such domains. Firstly, assisting environmental sustainability by identifying patterns within its components. Machine learning techniques play an important role here in many ways. Discovering associations between environmental components and agriculture is one such topic. Secondly, improving the robustness of Artificial Intelligence applications on embedded systems. AI has reached our day-to-day life through embedded systems. The technical advancement of embedded systems made it possible to accommodate ML. However, embedded systems are susceptible to various types of errors, hence there is a huge scope of recovery systems for ML models deployed on embedded systems. Third, bringing the user communities of the entertainment systems across the globe together. Online streaming of entertainment has already leveraged ML to provide educated recommendations to its users. However, entertainment content can sometimes be isolated due to demographic barriers. ML can identify the hidden aspects of these contents which would not be possible otherwise. In subsequent paragraphs, various challenges concerning these topics will be introduced and corresponding solutions will be followed that can address those challenges.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

Big data applications have always suffered from the challenge of data sparsity [1]. It is one of the biggest curses of dimensionality. The curse of dimensionality states the phenomena that there are certain challenges that arises when analyzing high dimensional data; whereas they start to fade while working with low dimensional spaces [2]. Data sparsity can be generated for various reasons, depending on the domain. Big data tasks that involve sustainability are notorious for broad changes and variations that affect a large fraction of the records [3]. For example, weather patterns may affect large geographic regions, and there may not be training samples in one year that are good representatives for a different year. That means that even when tens of thousands of data points are available, the predictive quality can be low. Even worse, if the data are not structured carefully, it might lead to misleading interpretations. Machine data is one of the most impacted field of data sparsity. Sparse data should not be mistaken with missing values. Missing values are generally due to data collection error. However, data sparsity refers to zero values which are valid. For example, the telemetric data has 90% zero values, which means that the reading from the corresponding sensor was zero. However, there could be missing values in GPS data where the location could not be reported due to the weak GPS reception [4]. A movie recommendation system, like any other recommender system, suffers from data sparsity [5]. For example, the MovieLens 100K dataset has a sparsity of 0.936953, which means that more than 93% of the data space is empty. It increases exponentially with the dimensions. The larger versions of ML datasets have even higher ratio of sparsity. Additionally, we consistently bring in more and more data sources to assist the predictive power of a model. Which relentlessly fuel the problem to a larger extent. There are many techniques described in the following chapters to deal with data sparsity based on the nature of the data

When various data sources are connected to bring more predictive power, systematic correlations are introduced [6]. Consider the case of predicting the productivity of agricultural land based on weather data. Weather patterns often extend over several counties or even multiple states [7]. That means that there are systematic correlations among precipitation, temperature and agricultural outcomes that affect most or all of those in the same region. In this setting, it is important to examine what large sets of data do provide, even when they do not sample the input and output

space well. When multiple dependent attributes are available, their correlation with the independent attributes stands out as especially information rich. In the case of Convolutional Neural Network, certain parts of the network build associations among themselves. Discovering these patterns can significantly improve the performance to rebuild the network in case of error. Similarly, a movie recommendation engine builds communities from the standpoint of watch list. Reviewers create clusters that are well separated due to various barriers. Discovering these communities can assist to induce interactions among these communities, which in turn can lead to expansion of reviewers' preferences.

The task of discovering association among components is affected by the mixing of effects. In other words, there could be confounding variables whose importance cannot be separated. There are multiple factors within an environment that make a compound effect on the agricultural components. However, the degree of impact in isolation is often unknown. This is different from controlled experiments where the settings of the experiment can be fixed to measure the impact in isolation. That is not applicable in most of the application in environmental sustainability. Additionally, machine data is collected using communication channels that are exposed to harsh environments and the quality of the data cannot be guaranteed most of the time. Proper precautions need to be taken to assure the robustness of the applications. Like any other recommendation system, a movie recommendation system is dependent on human behavior which is unpredictable. For example, A movie might get good ratings by reviewers from a certain region due to some trending event in that region. Social media mining can bring in some context, at the cost of increasing the dimensionality of the problem and introducing noise. Identifying universal latent features such as genre can keep the solution neat as well as robust.

Traditional linear prediction approaches are not affected by the presence of multiple dependent attributes, because each prediction task can be equivalently considered as a separate task. However, the availability of multiple dependent attributes does affect the overall strength of the relationship between independent and dependent attributes, even in canonical correlation analysis [8] , which is based on a linear model. For decades, the main importance of canonical correlations was on the development of special parametric approaches, since the computational and data collection involved was prohibitive. Now that sustainability questions motivate a broader use, limitations also become apparent: Conventional matrix-bases techniques work well for a small number of at-

tributes, but numerical instabilities prevent the techniques from being useful with the generality that is needed in the new contexts. The proposed approach in Chapter two is based on the clustering of dependent and independent attributes. Clustering depends much less on dimensionality than matrix-based techniques do. Clearly, even when different algorithms are used on the same data, results are not expected to be identical, and the proposed algorithm is based on the assumption that the differences in clustering between dependent and independent attributes can be substantial. The potential of the method rather derives from examining the overlap between clusters statistically, over all combinations of clusters.

A single dimensional outcome can often be explained in many ways. Adding more input dimensions makes this problem rather more severe, as all data points have a similar distance from all others in high dimensions. The correlation between input attributes and output attributes is different for the combined problem than for the combination of univariate ones. When multiple output attributes are available, using their full multidimensional space may counteract this problem, because it becomes less likely that neighbors in the input space will also be neighbors in the output space. In other words, the chance of accidental co-occurrences decreases.

The application of big data driven products was limited to powerful computers and cluster of distributed systems. With the surge of computational power and efficient energy consumption management on embedded devices, embedded processing has grown exponentially during the last decade. In particular, computer vision has become prevalent in real-time embedded systems, which have always been a victim of transient fault due to its pervasive presence in harsh environments. Computer vision in embedded systems is called Embedded vision. Like every other CV domain, Embedded Vision is also dependent on Convolutional Neural Networks (CNN), which are popular in this domain. It has shown unmatchable success at the cost of high volume and complex topologies of the necessary networks. Due to the sensitive nature of these applications and the vulnerability of the hardware, embedded vision suffers from the performance of software-based recovery model. Interestingly, there are certain patterns on the bit-level storage of the deployed models that govern the performance of these models. Associations among these bits were recognized and a stochastic recovery model is built. The third chapter demonstrates the experiments to establish the effectiveness of this model.

The fourth chapter of this dissertation focuses on the associations among the separable communities within review space on movie recommendation systems. As the internet keeps accommodating more and content for users, the relevance of content for users has become crucial. On the one hand, more content expands the options for users. On the other hand, it increases the complexity of finding the relevant items for users. In consequence, the task of recommendation engines has become more and more difficult. The recommendation systems available today can be broadly classified into three major categories: Collaborative-Filtering, content-based-Filtering, and a hybrid of the first two. In the hybrid approach, the content-user similarity matrix is built using both the content of the movies as well as the rating of the movies in common. The feature extraction based on the content of the movie is done by content-based techniques. The weights of the extracted features designate the importance of the corresponding feature to the user. Like collaborative-filtering, the similarities between users are calculated. However, the weights of the features are used in place of user ratings to calculate the similarity. This approach has lots of features to calculate the similarities between users. Hence, it deals with the issue of users having an insufficient number of rated movies. Moreover, the extracted features are capable of capturing users' preferences outside of their usual environment. This is a considerable uplift from content-based-filtering. However, we still need a significant number of users that have rated a movie for the recommendation. Similar to collaborative-filtering, it is not possible to recommend a brand-new movie unless there is a user who has rated it.

This dissertation contributes to various topics related to pattern recognition and associations attempting to solve problem statements on diverse domains.

- Multiple novel similarity metrics for multivariate datasets have been proposed. The usability of the metrics was established for time series datasets as well.
- A stochastic recovery model is introduced based on the bit-level associations of pretrained Convolutional Neural Network.
- A genre similarity-based Movie recommendation engine is proposed. This work is presently in progress.

This dissertation consists of five chapters. Each of the subsequent chapters originated from a published or a submitted paper. These papers were based the research studies as part of the doctorate program. The following is a brief overview of each chapter.

- Chapter 2 proposes two novel similarity metric for multivariate datasets based on collective proximities of the data points in two datasets. The algorithm was implemented for experiments to improve environmental sustainability.

  - Denton, A.M. and Roy, A., 2017, December. Cluster-overlap algorithm for assessing preprocessing choices in environmental sustainability. In 2017 IEEE International Conference on Big Data (Big Data) (pp. 4212-4220). IEEE.
  - Roy, A. and Denton, A., 2019, May. Nearest-neighbor-intersection algorithm for identifying strong predictors using high-dimensional data. In 2019 IEEE International Conference on Electro Information Technology (EIT) (pp. 416-421). IEEE.

- Chapter 3 proposes a software-based recovery model for Silent Data Corruption on deployed Convolutional Neural Network. The application is focused on improving the robustness of embedded vision.

  - Roy, Arighna and Ludwig, Simone A. 'Recovery Algorithm to Correct Silent Data Corruption of Synaptic Storage in Convolutional Neural Networks'. 1 Jan. 2020 : 177 – 187.

- Chapter 4 discusses a content-based movie recommendation engine that leverages genre similarities to deal with the Cold-Start problem.

  - Roy, A., Ludwig, S.A. Genre based hybrid filtering for movie recommendation engine. J Intell Inf Syst 56, 485–507 (2021).

2. NON-LINEAR ASSOCIATIONS BETWEEN MULTIVARIATE DATASETS FOR EFFECTIVE PREPROCESSING

Precision agriculture benefits from the insights drawn through big data analysis. In this paper, we discuss the challenges associated with big data analysis and its relevance to precision agriculture. Insights generated from the associations between the environmental and agricultural attributes can be used for various preprocessing techniques. These techniques can enable better weather decisions to improve yield productivity. Additionally, we propose two novel algorithms to capture the non-linear relationship between two sets of attributes without having to depend on function approximation or spectral decomposition of the data matrix. The core problem statement can be defined as estimating the association between two vectors of random variables through paired observations. The majority of the available techniques to solve this problem have been founded upon Canonical Correlation Analysis. The limitation of the existing methods is discussed in the related work section.

Big data has unique characteristics that are not shared by traditional datasets [9]. That leads to unique challenges both computational and statistical [10]. The predominant features are categorized by massive sample size and high dimensionality [11]. However, there are other unique characteristics of big data such as heterogeneity [12], complex structure [13], and sparsity [14] to name a few. Modeling this intrinsic diverse nature of big data requires methods that are tailored to specific use cases [15] [16]. The curse of dimensionality comes with the high dimensionality of the data. Due to the curse of dimensionality, big data faces challenges such as noise accumulation [10] and spurious correlation [17]. As the number of dimensions increases, the volume of the space increases exponentially. Analyzing such data requires simultaneous estimation or testing of several parameters [18]. Errors associated with the estimation of these parameters accumulate when a decision rule or prediction rule depends on these parameters. Such a noise accumulation effect is severe in high-dimensional data [19]. Consequently, high dimensionality brings in spurious correlation due to the fact that many uncorrelated random variables may have a high sample correlation coefficient

in high dimensions [17]. Spurious correlation in turn can cause misleading inferences. It is the existence of a correlation between variables "unintentionally" as well as due to "high-dimensionality" [19]. For example, in weather forecasting, two sub-regions do not have similar weather conditions, but they show significant associations by chance in presence of numerous weather variables that are highly correlated with each other. In a contrary scenario, consider an example of predicting the agriculture production under climate change due to carbon dioxide emission [20] [21]. This data pattern can be extended over several regions or even for long time periods (span of years). It can be found that there exists a causal relationship between agriculture productivity and carbon dioxide emissions [22] for those regions or time periods. So due to the curse of dimensionality of big data, there can be multiple confounding factors that can accumulate, and the insights, to some extent, can be biased to the context or event [23] [24].

Big data applications are inherently different from controlled experiments [25] where causal inferences can be established through directional relationships. However, due to the heterogeneity of the data, it is very risky to establish a causal relationship for big data applications. Bidirectional relationships [26] [27] have fewer strong assumptions and fit better to big data applications. For the applications where variables contain confounding relationships, the supervised learning approach may not be useful [28] due to the presence of transitive relationships among the variables. Such relationships are predominant in features considered for environmental sustainability and precision agriculture applications [29]. However, capturing the associations among the variables (dependent or independent) leads to meaningful inferences [30] [31].

Another challenge with big data application is the degree of complexity within the variables. Nonlinear relationships between the attributes are prevalent in big data applications [32]. The presence of nonlinear relationships among the attributes becomes a major challenge for the function approximation approach.

Figure 2.1 illustrates the problem. Consider two vector space representations of the same objects. The left two panels represent one vector space, while the ones on the right represent the same objects in a different vector space. We select each object and look for the nearest one or more neighbors in the left vector space and also for the nearest one or more neighbors in the right vector space. In the figure, one object and its nearest neighbor in the left vector space are highlighted. If the information in the vector spaces is related, we expect that in some cases the nearest neighbors

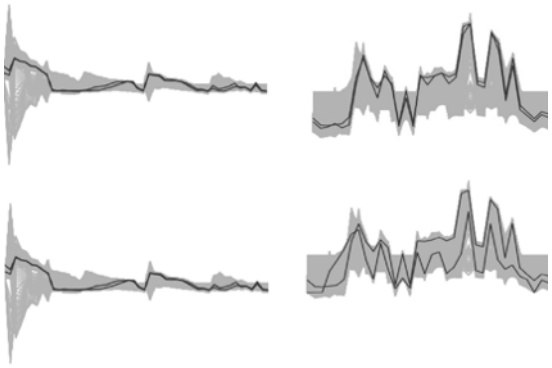


Figure 2.1. Schematic to illustrate the problem. The top figure shows a set of time series in gray. One of the time series and its nearest neighbor are shown in black. The two panels on the left side correspond to one vector space, and those on the right side correspond to another. For the to panel the objects that are the nearest neighbors on the left side are also nearest neighbors on the right, while that is not the case for the bottom panels

in both vector spaces will overlap. Even if the vector spaces were completely unrelated, we would still expect to see occasional matches through random chance alone. Our algorithm is based on a comparison of the observed number of matches with the expected number. In the top row, the two highlighted neighbors in the left panel are also neighbors in the right panel, which supports that a relationship between the vector spaces exists. In the bottom row the two neighbors from the left panel are not close in the vector space of the right panel. This can happen even for related vector spaces, but we expect to have more matching neighbors in the case of related vector spaces than through random chance. The algorithm is applied to all objects and the results are aggregated.

We have extended the above assumption to build the clustering overlap algorithm. If two datasets have a strong relationship between them then the same data points are likely to form clusters in both datasets. In other words, we should expect to see a significant amount of overlap (of data points) between the clusters in both datasets. Analogically in Figure 1, the top two diagrams show the rows that belong to one cluster in the left panel also belong to one cluster in the right panel. Alternatively, the bottom two diagrams show otherwise when the two panels are not associated.

2.1. Related Work

Quantifying associations between two sets of multivariate datasets is a century-old problem. It evolved from capturing linear association between two variables which is known as Pearson

Correlation Coefficient [33]. Fifty years later, Hotelling extended it to the multivariate problem space with the Eigenvectors of the cross-correlation matrix [34] and is known as Canonical Correlation Analysis. The objective is to find the linear combination of the variables that shows the maximum correlation between the two sets of variables. Most of the parametric significance testing tools available for two sets of variables can be considered as a special case of Canonical Correlations (CCA) [35]. Hence, it provided a significant contribution to the development of parametric statistical techniques in the pre-big-data era [36]. Another field of data science that benefited from CCA is multivariate regression. With the abundance of available data, the prediction tasks are overwhelmed by the curse of dimensionality. Variable selection or dimensionality reduction has been a critical part of the forecasting tasks for better interpretation of the results [37]. CCA played a significant role in [38]. Additionally, larger datasets came with computational complexity. Prior to the distributed computation era, it was a game-changer for many tasks.

Over the years, various extensions of CCA have been developed. For example, Liang [39] examined the effect of the regularization technique by applying Sparse CCA formulation on a multi-label classification task as a least-squares problem. Such methods made CCA applicable to large and sparse datasets. Multi-label Canonical Correlation Analysis is introduced to tackle multi-label annotations by Viresh et al. [40] for cross-modal retrieval.

One of the major limitations of CCA lies in its assumption of the linear relationships between two sets of variables [41]. A linear relationship between two sets of attributes might not be adequate for a lot of analysis tasks [42]. KCCA is an extension of CCA that overcomes this limitation [43]. KCCA first converts the data points to a Kernel-augmented representation and then extracts the canonical variates in the transformed Hilbert-Space [43]. The Gaussian density function is a common choice of the Kernel function for this task. Like CCA, it relies on Spectral decomposition to extract the Eigenvectors. These Eigenvectors are the Canonical variates. Su-Yun et al. [42] performed Bartlett's test for the KCCA significance test on both synthetic data and handwritten digits. However, the Bartlett's test suffers from sensitivity towards the non-normality of the data [44]. Additionally, KCCA is computationally expensive due to the kernel-augmented data transformation [45].

With the rise of distributed computation, Neural Networks gained popularity in the machine learning communities as well as data science communities. Galen Andrew et al. utilized a neural

network to build an extension of CCA, namely Deep Canonical Correlation Analysis (DCCA), to learn the non-linear relationship between two sets of datasets [46]. Like KCCA, DCCA is a non-parametric technique. However, unlike KCCA, it does not require computing inner-products of the data points which is computationally expensive. One major limitation of DCCA, like any neural-network-based tool, is the dependency on large sample sizes especially in the context of high-dimensional datasets.

Within the scope of this chapter, we have discussed how quantifying the association between environmental and agricultural attributes can benefit precision agriculture. Similar work has been done on soil sensors for better profitability from crop production and product quality improvement [47] [17]. Although yield-related attributes are the most useful metric in agriculture, the data collection process for the same is a rigorous process. Studies in [48] [49] [50] show that the vegetation index extracted from Satellite imagery is highly correlated with the yield data, and therefore can be alternately used to serve this purpose.

2.2. Method

We are considering two separate sets of variables for same observations and trying to find if the combination of one set of variables is correlated to the combination of the other set of variables. Consider two sets of attributes, X and Y; X has a number of attributes and Y has b number of attributes. The dataset contains n data points and both X and Y have paired sample values for those n data points. The ith data point in the X set is denoted as Xi and the paired data point for the Y set is denoted as Yi. In addition, the mth attribute in the X set is denoted as X(m). Hence, the mth attribute of X for the ith data point is denoted as Xi(m). For two given instances i and j, the distance between the data points in the vector space of X is DX(i,j); and the equivalent one for the Y dataset is DY(i,j). The underlying assumption of our algorithm is that DX(i,j) and DY(i,j) form a pattern collectively if X and Y have a significant association. To put it into perspective, say X is the daily rainfall and Y represents multivariate plant quality attributes. If there is a strong association between rainfall and plant quality then we expect to see the locations with similar rainfall patterns having similar plant quality.

Consider a data set with two sets of variables, $x = (x_1, x_2, \ldots, x_m)$ and $y = (y_1, y_2, \ldots, y_m)$ with the length of m and n, respectively. The ith sample of the data set would be $(x_{i1}, x_{i2}, \ldots, x_{im}, y_{i1}, y_{i2}, \ldots, y_{in})$.

2.2.1. k-NN Intersection Algorithm

The idea of the k-NN Intersection algorithm is that if an instance has k nearest neighbors in one vector representation, and the same nearest neighbors in the other, the two vector representations are likely to be related. Regardless of how many of the neighbors overlap, the p-value of the observation can be calculated. A generalization to evaluating the significance based on a combination of all instances is straightforward.

In the k-NN Intersection (NNI) Algorithm we consider neighborhoods consisting of the nearest neighbors to an instance l in both vector spaces:

$$(U_l^a \mid \forall m \in U_l^a, n \notin U_l^a, d(x_l, x_m) \leq d(x_i, x_n) \wedge |U_l^a| = k_{nn} \tag{2.1}$$

$$(U_l^b \mid \forall m \in U_l^b, n \notin U_l^b, d(x_l, x_m) \leq d(x_i, x_n) \wedge |U_l^b| = k_{nn} \tag{2.2}$$

For the calculation of the expected overlap, the instance that is characterized by x_l and y_l is excluded from the consideration, and the probabilities are calculated over $N-1$ points. Each of those instances has a probability of $k_{nn}/(N-1)$ of being in the subset that's defined by being x_l and, independently, a probability of $k_{nn}/(N-1)$ of being in the subset that's defined by being y_l. Overall, the expected number of shared neighbors within any pair of unrelated neighborhoods is $\frac{k_{nn}^2}{N-1}$.

The number of shared elements of neighborhoods is furthermore aggregated over each of the possible instances l resulting in a total of

$$n_{\text{expect}} = E\left(\sum_l |U_l^a \cap U_l^b|\right) = \frac{k_{nn}^2}{N-1}$$

if vector attributes **A** and **B** are unrelated.

After the aggregation, the expected values are large enough that the Poisson distribution that is used for comparison can be approximated by a Gaussian distribution. The number of standard deviations above the mean, z is reported, with $z > 2$ considered an indication that the distributions are related

$$z = \frac{\left|\sum_l |U_l^a \cap U_l^b| - n_{\text{expect}}\right|}{\sqrt{n_{\text{expect}}}} \tag{2.3}$$

2.2.2. Cluster Overlap Algorithm

The foundation of Cluster Overlap algorithm is similar to that of the k-NN Intersection algorithm. When we cluster the data points in the vector spaces of the attributes of X and Y separately then we expect to see a significant overlap between the data points residing in the same cluster. However, there can be several such overlaps by random chance. We need to test the significance by comparing the observed and the expected overlap counts for every combination of the cluster pairs between the datasets.

$$\chi^2 = \sum_{i=1}^{k_{\text{clust}}} \sum_{j=1}^{k_{\text{clust}}} \frac{\left(|C_i^a \cap C_j^b| - \frac{|C_i^a|\,|C_j^b|}{N} \right)^2}{\frac{|C_i^a|\,|C_j^b|}{N}} \tag{2.4}$$

where C_i^a and C_j^b are the members of clusters a_i and b_j respectively, i.e. $|C_i^a \cap C_j^b|$ is the number of data points in the intersection of the two clusters, and its expected value, assuming independence between the two clusters, is $\frac{|C_i^a|\,|C_j^b|}{N}$

The above statistic should follow a χ^2 distribution with a degree of freedom $f = (k_{\text{clust}} - 1)^2$, assuming the independence between any two clusters. With a large enough sample size, we can approximate that with a Gaussian distribution with df mean and $\sqrt{2f}$ standard deviation.

$$Z = \frac{\chi^2 - (k_{\text{clust}} - 1)^2}{\sqrt{2}(k_{\text{clust}} - 1)} \tag{2.5}$$

We identify a relationship to be significant for the above test statistic to be bigger than 2.

2.2.3. Limitations of Existing Methods

Normality of the data: Although canonical correlations do not limit to multivariate normal data sets, it definitely works better for such data. For KCCA, Bartletts test of independence has been used to kernel augmented data [42]. Bartlett's test is sensitive to departures from normality.

Linearity of relationship: Canonical Correlation analysis assumes linearity relationships among and between both dependent and independent sets of variables. It cannot capture the non-linear components of the relationship. This one drastically limits the usefulness of this analysis while dealing with time series data sets which have a nonlinear relationship between themselves. KCCA relaxes the assumption of linearity, at the cost of the assumption of a smooth population regression function which is much weaker than linearity assumption [51]. But, nonlinear represen-

tation is limited by the fixed kernel [52]. KCCA requires an inner product. Because of which the computation cost scales poorly with the data.

Homogeneity of variance: CCA assumes a stable distribution of variance throughout each dataset. Bartlett's test for KCCA also assumes the homogeneity of variance within each of the dataset. That is often not the case in real life dataset.

Multicollinearity: The performance of CCA suffers from Multicollinearity and singularity; Multicollinearity occurs when the independent variables are highly correlated. Singularity occurs when you have redundant variables. Although, this feature can be fulfilled by doing a preprocessing on both the sets of variables separately. Although KCCA is not affected by this constraint, the nonlinear mappings of the vectors may camouflage the dependencies between their original vector representations [53].

Dependency on large sample size: Like multivariate regression, CCA performs better for large number of rows. KCCA requires an even larger sample size because the data must supply the model structure as well as the model estimates.

Overfitting: KCCA is prone to over-fitting errors. Although it can be resolved by regularization methods.

2.3. Experimental Results on Time-Series Datasets

The algorithms were implemented in R. The experiments were done on a 2.6 GHz Intel Core i7. We have considered various approaches and data sets to portray the consistency of our claim. Our experiments include various values of dimensions, randomly sampled instances and model hyper parameters to conclude our argument.

The motivation for this work is to establish the strength of relationships in data with multiple independent and dependent attributes. Making definitive statements on what results are to be expected is inherently difficult for such complex data sets. In addition, they don't provide the flexibility of varying the number of attributes flexibly. For that reason, experimental data are constructed from real time series data from the UCR Time Series Repository [54], but for which we know what results are expected. We use that as a proof of concept for our model.

First, we have optimized the k-Nearest Neighbor Intersection algorithm based on effectiveness and efficiency. Thereafter we compare k-Nearest Neighbor Intersection algorithm with the existing algorithms based on effectiveness and efficiency. The two most popular methods are con-

sidered for comparison which are Canonical Correlation Analysis (CCA) and Kernel Canonical Correlation Analysis (KCCA), to set the ground for both linear and non-linear relationships. We have demonstrated how sample size and the length of the time series impact the performance of the algorithms. While demonstrating the effect of sample size on the performance, the Accuracy is averaged over multiple length of time series to remove any bias due to specific length of time series. Following are the length of time series considered: 30, 40, 50, 60, 80, 120. Similarly, while demonstrating the effect of length of time series on the performance, the accuracy is averaged over multiple sample size. And, following are the sample size considered: 50, 100, 200, 500, 1000.

2.3.1. Data Preparation

For the evaluation on time series data we split time series into two halves. The first and second half are expected to represent related sets of attributes. As a control experiment, the second half of the time series is randomly shuffled. After the shuffling, the first and second halves are not expected to be related, i.e. two instances that are close in the first half, are not normally expected to be close in the second half by random chance.

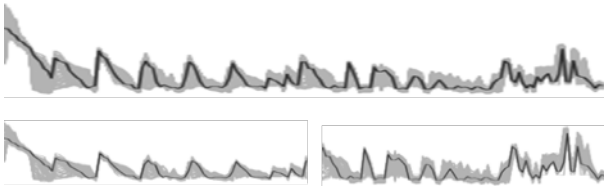


Figure 2.2. Schematic diagram to explain the data preparation

Figure 2.2 explains the data preparation visually. The top figure represents a randomly chosen time series dataset from UCR repository with the time points across the X axis. Thereafter we choose half the time points from beginning for the first dataset and the rest for the second dataset. The order of the time points is maintained to display the match. Thereafter, the time points were replaced by the concept of columns to fit it into linear algebra theory. One data point is highlighted in the original time series (top figure) and in both the transformed datasets (bottom figures). To modify the length of the dataset for the interest of experiments, we select time points from fixed interval. This can be compared with systematic sampling.

2.3.2. Terms Related to Performance

Let us explain the standard performance metrics in the context of our experiments. Evaluating a correlation analysis model has always been tricky. The amount of correlation among datasets really depends on the type of relationship the model tries to capture. To standardize the evaluation, we transformed the problem into a binary classification problem. The label of the classification dataset is whether or not the instances on the right side of the pair are shuffled or not; or in other words whether the pair is supposed to be correlated or not. Suppose, one time-series data set is split into two halves and those are predicted to be related in our experiment. That is a True Positive case. If those are predicted to be not related then the experiment is considered to be the False Negative. This is based on the assumption that splitting a time series data in halves generate a pair of related datasets. If the right half instances are randomly shuffled and then the pair is predicted to be not related in our experiment then it is a True Negative case otherwise False Positive. If execution gets unexpected exceptions then we tag it as a failed experiment. We have included these experiments in the denominator of the equation of accuracy.

2.3.3. Evaluation Metrics

The effectiveness is measured using Accuracy and F1 score. And the efficiency is measured using the execution time. We have used Accuracy for showing the overall performance of the model depending on various factors. In general, Accuracy is not a robust evaluation metric for classification tasks due to its inability to work with imbalanced data. For a binary classification experiment, there are four usual options for the result of a trial: true positives, false positives, true negatives, and false negatives. In this experiment, we have another option, a failed trial. It happens when the algorithm runs into a numerical or a computational error during the trial. There can be test cases that fall into this category and it makes sense to capture them in the evaluation metric of the effectiveness of the algorithm. We have added them in the denominator of accuracy. We prepared a balanced experiment data which solves that problem. F1 score has been used to provide the optimal performance of the model for a design of experiment. F1 score indicates the Harmonic Mean of Precision and Recall; hence it includes both Precision and Recall into a single measure.

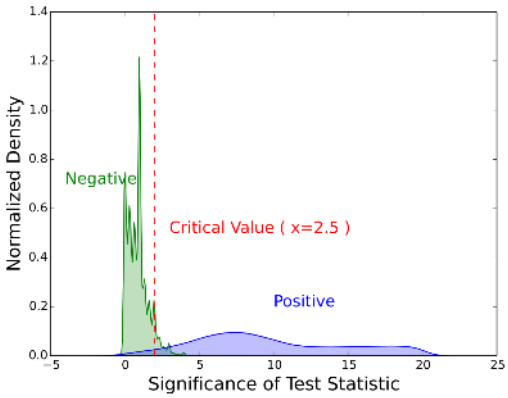


Figure 2.3. Distribution of test statistic for positive and negative test cases

2.3.4. Optimization of the Significance Value of the Test Statistic

In this section we will discuss the process of regulating the optimal significance threshold of the test statistic we defined in the previous chapter. Figure 2.3 portrays the distribution of test statistic values of both negative (null hypothesis) and positive (alternate hypothesis) cases overlaid.

Our goal is to estimate the empirical critical value which yields the best possible tradeoff between significance level and power of the statistical test. 6.5% significance level can be achieved with a critical value of 2 with the power of the test as 97.3. A critical value of 2.5 produces a more conservative model with a significance level of 2.9% and power of 97.2%. So, we can reduce a significant amount of false positive rate (FPR) by compromising negligible amount of power or allowing negligible number of False Negatives. Beyond 2.5 the FPR doesn't reduce much but the power starts to reduce significantly. We decided to use 2.5 as the critical value of the significance of the test. To justify the trade-off between significance level and power, we can use F1-score. Critical values 2, 2.5, 3, 4 yield F1-scores of 0.95, 0.97, 0.97, 0.96 correspondingly. So, we can choose either 2.5 or 3 as the critical value.

Table 2.1. Various combinations of test parameter values

Critical Value	**Significance Level**	**Power**	**F1**
2	6.5%	97.3%	0.95%
2.5	2.9%	97.2%	0.97%
3	0.9%	95.3%	0.97%
4	0.5%	92.7%	0.96%

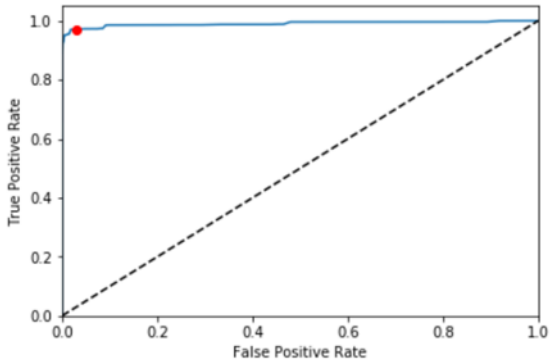


Figure 2.4. Receiver Operating Characteristic (ROC) chart for the test statistic values

Figure 2.4 visualizes the tradeoff between power (TPR) and significance level (FPR) of the test statistic values as explained above. The elbow of the ROC curve is considered to the optimal critical value. From the above diagram, it's very difficult to find the elbow. The red dot in the diagram displays the critical value of 2.5 (refer to Table **??**) and we can see that beyond that point, the power doesn't increase much at the cost of FPR.

2.3.5. Dependence of Performance on k

We have conducted multiple experiments on the number of nearest neighbors considered for the k-Nearest Neighbor Intersection algorithm to find the optimal value of the hyper parameter. Following are the values we have considered for k: 1, 3, 5, 10 as part of our experiment. In one set of experiments, we have monitored how the sample size effect the accuracy of the algorithm

for each of the values of k. For each sample size, the accuracy is averaged over multiple length of time series (30, 40, 50, 60, 80, 120). Figure 2.5 shows the effect of sample size on the effectivess of the algorithm. In the other set of experiments, we have monitored how the length of time series effects the accuracy of the algorithm for each value of k. For each time series length, the accuracy is averaged over multiple sample sizes (50, 100, 200, 500, 1000). Figure 2.6 shows the result for the same.

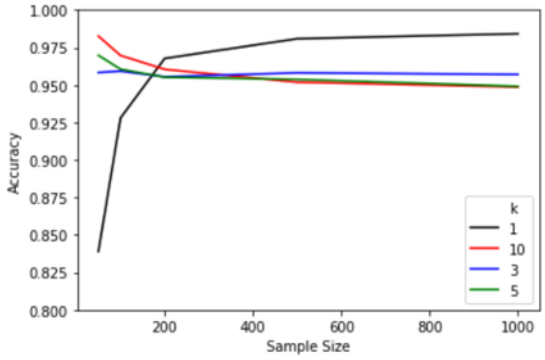


Figure 2.5. Accuracy ((TP + TN) as a percentage of all tests) for each of the k, depending on the sample size, i.e. number of rows considered

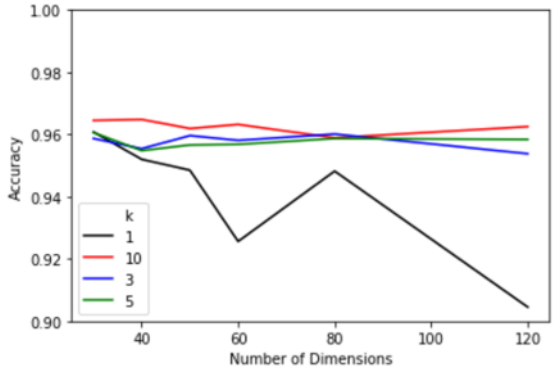


Figure 2.6. Accuracy ((TP + TN) as a percentage of all tests) for each of the k, depending on the number of dimensions, i.e. length of the time series considered

The main motivation of this algorithm is to be able to find the relation where the existing algorithms fail to perform. Both CCA and KCCA perform poor for low sample size or high dimen-

sions. NNI performs poor in those scenarions too when $k = 1$; So, we discard $k = 1$. In Figure 2.6, $k = 10$ shows the best performance for NNI. But Fig. 2.9 and 2.10 shows that the execution time of NNI increases with k, especially for higher sample size. So, the optimal value of k is considered as 3. We have used $k = 3$ for rest of the experiments.

2.3.6. Dependence on Sample Size

One of the major challenges most of the data science problems face is the sample size. Specially with high dimensional data, the need of more data points grows exponentially due to curse of dimensionality. In this section, we are interested in investigating the impact of sample size on the performance of the model.

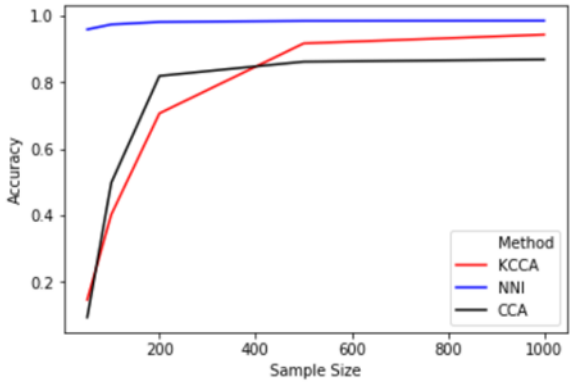


Figure 2.7. Accuracy ((TP + TN) as a percentage of all tests) for each of the algorithms, depending on the sample size, i.e. number of rows considered, with $k = 3$ for k-NN Intersection Algorithm, Canonical Correlation, Kernel Canonical Correlation

Fig. 2.7 demonstrates that NNI outperforms both CCA and KCCA for all the sample sizes. The accuracy is really poor for CCA and KCCA when the sample size is small. Whereas NNI performs almost as equally good for small sample size as for larger sample sizes. The main reason behind such poor result of CCA and KCCA for low sample size is the failed experiments. Those fail to produce any result for low sample size, especially for longer time series. For each sample size, the accuracies for various length of time series were averaged. And we have included the failed experiments in the denominator of the accuracy which leads to lower accuracy. That's why we chose accuracy to evaluate the performance as it includes the failed experiments.

Also, a poor performance of CCA and KCCA for smaller sample sizes indicates a sign of overfitting as it gets better with larger sample size. We always don't have large enough sample size to maintain this performance. NNI significantly outperforms the existing ones for sample size lower than 500 whereas keep outperforming them (with lower margin) for larger sample size.

2.3.7. Dependence on Time Series Length

The best suitable use cases for NNI are the ones that require to capture relationships between two vector representations of different lenghts, KCCA and CCA fail to produce any result when the difference in length is high. In our experiments, we have kept the length of one of the vector representations fixed to be 30 and varied the length of the other one.

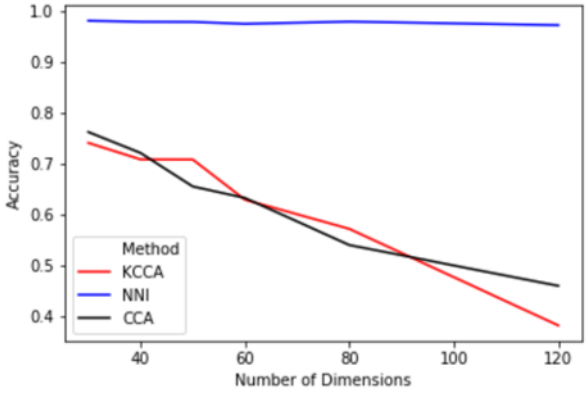


Figure 2.8. Accuracy ((TP + TN) as a percentage of all tests) for each of the algorithms, depending on the number of dimensions, i.e. time series length, with $k = 3$ for k-NN Intersection Algorithm, Canonical Correlation, Kernel Canonical Correlation

Figure 2.8 demonstrates that the accuracy of CCA and KCCA drops sharply when the length of the time series increases because that increases the difference between the time series lengths (the length of one of the time series 30). NNI performs equally well for all the length of time series. KCCA performs worse than CCA for longer time series as expected.

2.3.8. Efficiency of the Algorithm

Efficiency plays an important role while working with big data analytics. We have investigated the efficiency of NNI based on various levels of hyper parameters depending on experimental parameters. We have also demonstrated a comparative analysis of NNI with the existing algorithms.

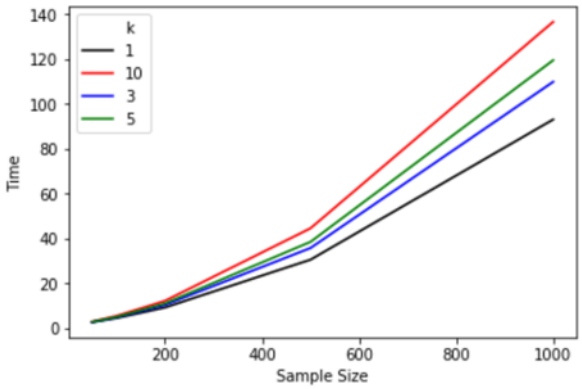


Figure 2.9. Runtime of NNI for each of the k, depending on the sample size, i.e. number of rows considered

We have showed how the efficiency of NNI scales with sample sizes using time series of length 120 in Figure 2.9. We have also checked how the efficiency of NNI scales with time series length using sample size of 50 in Figure 2.10.

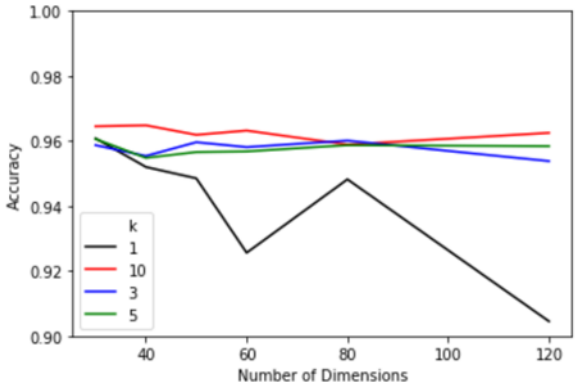


Figure 2.10. Accuracy of NNI for each of the k, depending on the number of dimensions, i.e. time series length.

Figure 2.11 shows that the efficiency of CCA drops exponentially with both sample size and length of time series; whereas the efficiency of NNI drops with sample size but still it is faster than CCA. And the efficiency remains almost constant with length of time series. KCCA is the non-linear version of CCA and is slower than CCA. We have not included KCCA in the comparative analysis.

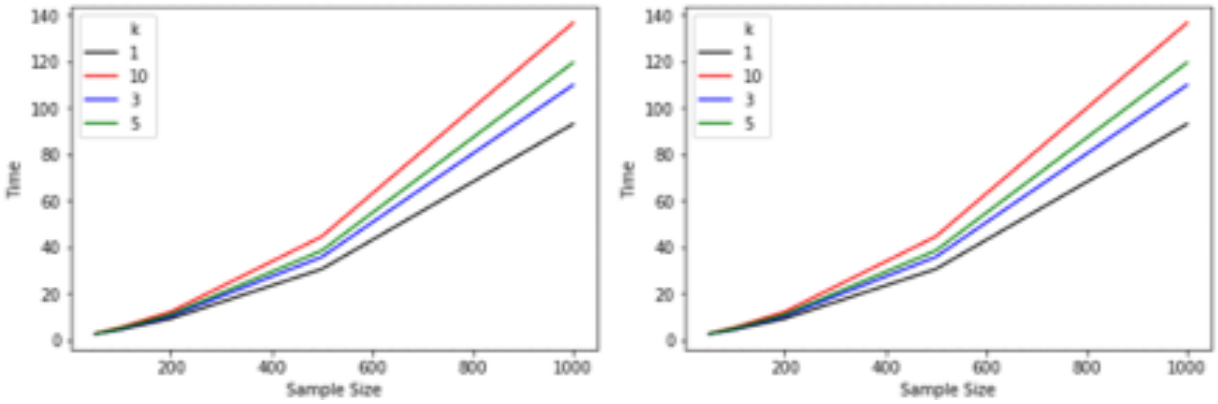


Figure 2.11. Execution Time for each of the algorithms, depending on the sample size, i.e. number of rows considered (left) and dimension, i.e. time series length (right)

2.4. Experimental Results on Environmental Dataset

One of the major objectives of this paper is to contribute to advanced precision agriculture techniques. This is possible to extract insights from the association between environmental and agricultural properties. These insights can lead to effective decision-making systems in the context of precision agriculture.

To establish the effectiveness of relying on the association between environment and agricultural attributes, we have considered rainfall time series and Beetroot-specific crop properties for around 4,000 coordinates. We collected the daily rainfall data for these coordinates for 5 years (2007 to 2011). The crop properties were also collected for the same coordinates for the 5 years. After several preprocessing steps, we quantify the level of relationship between these two datasets. Interestingly, the rainfall dataset is time-series data whereas the crop attributes dataset is cross-sectional data. Each year's data for a certain coordinate is independent of each other as they represent different rows within the system. In other words, each row in the final dataset represents a coordinate-year combination. Hence, we have around 20,000 rows for the dataset. Each row of the rainfall dataset is paired with the same row index on the agricultural dataset for the corresponding coordinate. More information on the dataset can be found in [55] [56] [22] [23]. Rainfall data is preprocessed as the first time point is the plant date in the corresponding coordinate and daily rainfalls for the next 180 days were considered.

Like any data science problem, we started our experiments led by some exploratory analysis on the input data.

2.4.1. Rainfall Data

One of the common problems in big data is data sparsity. We tested for the same in our rainfall dataset. The histogram in Figure 2.12 shows the number of days in a location when there was no rainfall since the plant date. Clearly, the data is very sparse here due to a lot of days without any rainfall.

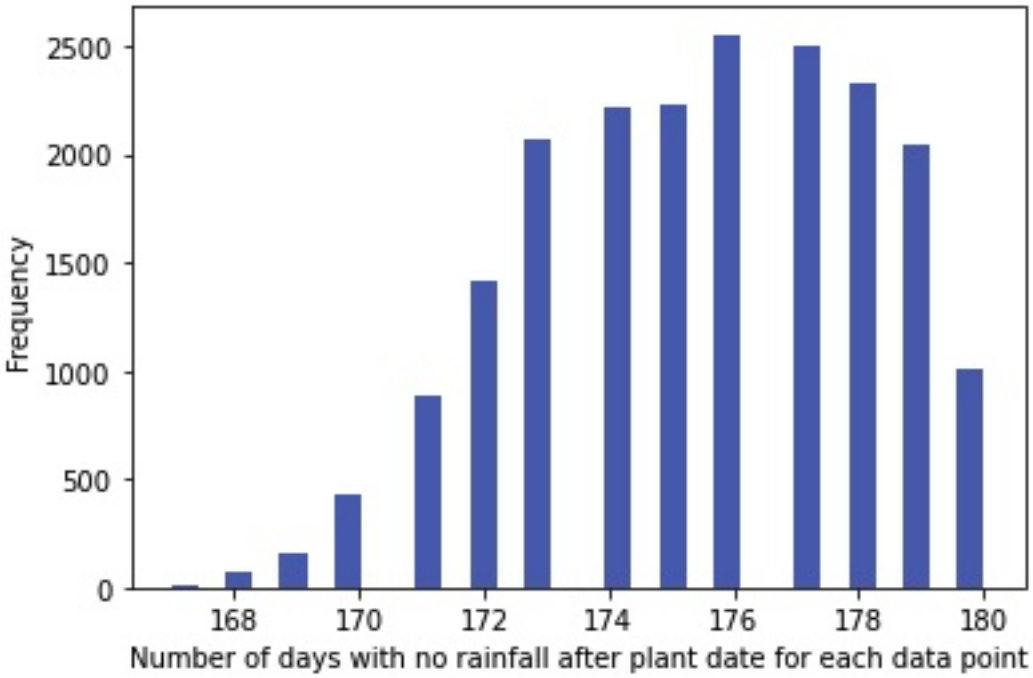


Figure 2.12. Number of days without any rainfall in a location

Figure 2.13 shows the distribution of non-zero rainfalls in a boxplot. We see a highly right-skewed distribution, even when there was any rainfall. This means that most of the days when there was rain, it was below 0.5 inches for daily aggregation.

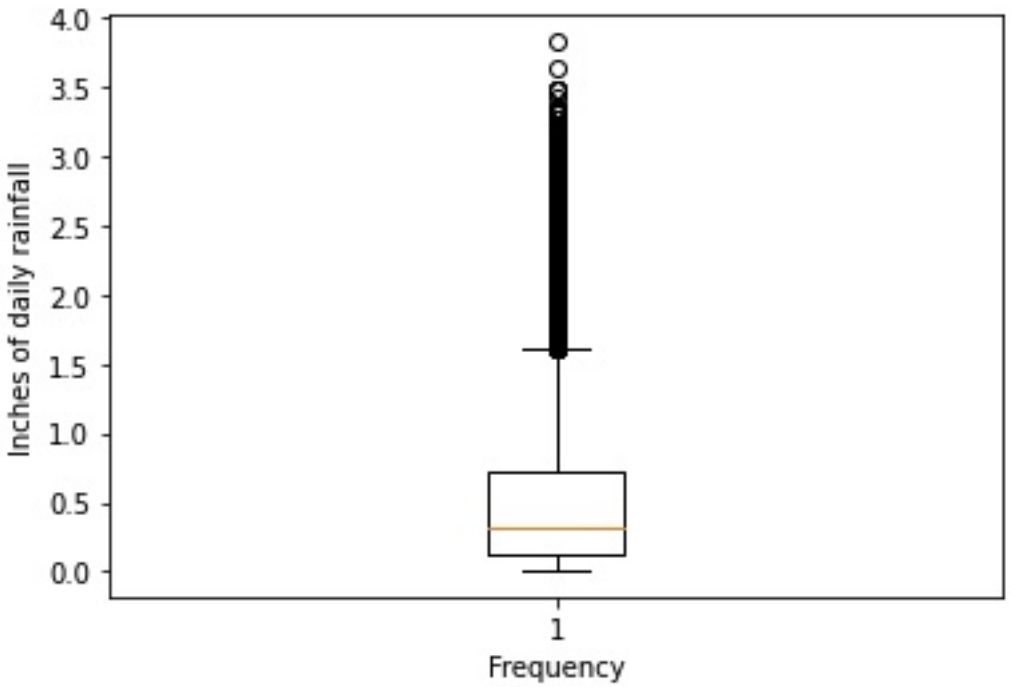


Figure 2.13. Distribution of non-zero rainfalls

2.4.2. Agricultural Dataset

For agricultural attributes, we considered yield, sugar content, and sugar-lost-in-molasses. The last 2 attributes are specific to beetroot and they indicate the quality of the crop. All three attributes are highly skewed and only sugar content is left-skewed (check Figure 2.14). Although, most of the parametric function approximation methods do not depend on the assumption on the distribution of the independent attributes, skewed distribution makes the inference less reliable. Association estimation using Cluster-Overlap based method is less sensitive to skewness of the attributes compared to its parametric peers.

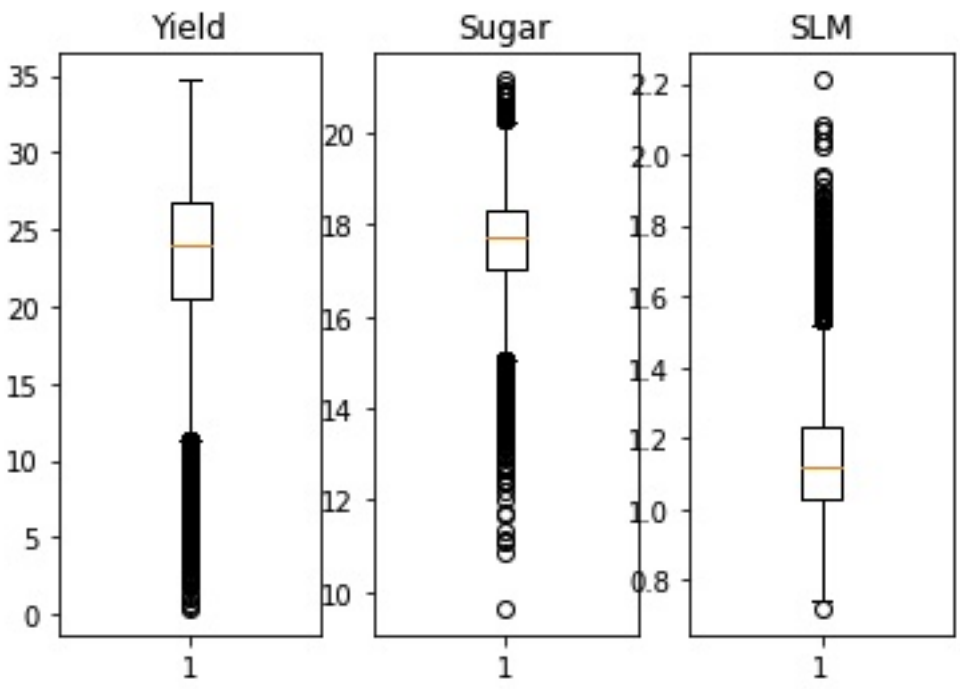


Figure 2.14. Boxplot on the distribution of the agricultural attributes

Figure 2.15 shows the pairwise Pearson correlations among the agricultural attributes. None of the agricultural attributes showed a significant correlation with any other two attributes. Although we do not observe much colinearity among the attributes, with increasing number of attributes finding such co-dependence becomes more likely.

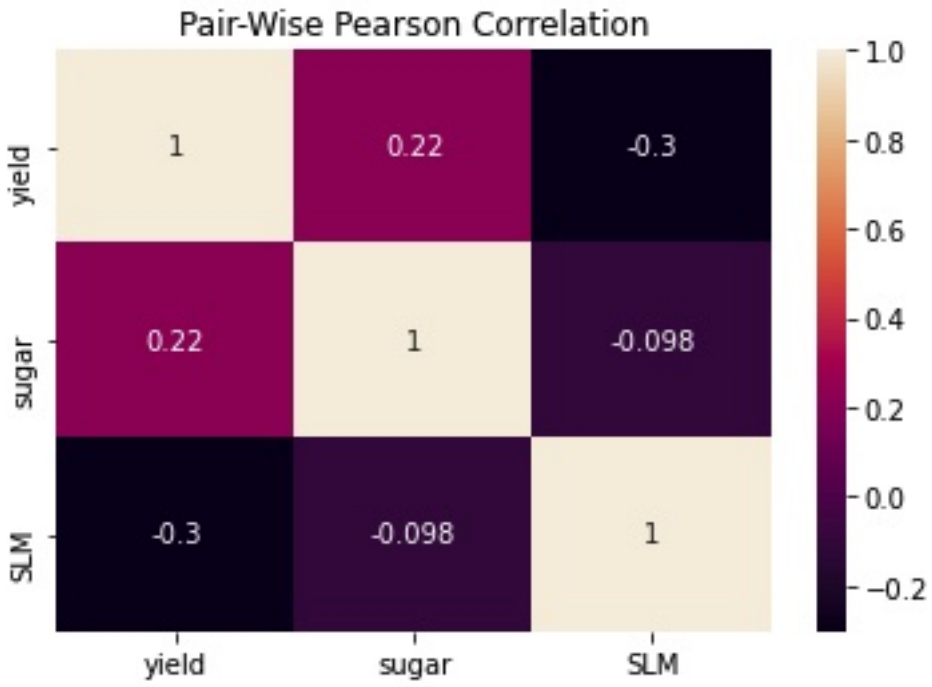


Figure 2.15. Correlation matrix for the agricultural attributes

2.4.3. Overview of the Design of Experiments

We have performed the experiment on a randomly chosen sample of a certain size to make one iteration. If the sample set could show a significant relationship between rainfall and plant quality, we consider that iteration to be a success. We repeat this process for 100 iterations to calculate the success ratio, which is the number of times the algorithm could detect a significant relationship between rainfall and plant quality. Hence the success ratio is bounded in (0,1). We have utilized the Cluster-Overlap (CO) algorithm and kept a threshold of 2 for the test statistic. We have also repeated the experiments for various cluster sizes (1-5) and averaged the success ratio for each experiment. Additionally, we have performed repetitions over preprocessing techniques (rainfall aggregation, rainfall thresholds, etc.) and various sample sizes were considered. These repetitions helped us to remove biases coming from certain control parameters. As the experiments are performed in an uncontrolled setup, these aggregations over repetitions were critical to establishing the results with as few biases as possible.

2.4.4. Experiments on Rainfall Thresholds

All these experiments were performed for various numbers of cluster centers (K as mentioned before). Also, the daily rainfall needed to be aggregated to create a series of various intervals (weekly, twice a week, once in two weeks, and monthly). For each of these aggregations, we have used both the maximum of the interval and the summation of the interval.

Figure 2.16 and Figure 2.17 show the strength of the relationship between the rainfall data and agricultural data for various preprocessing thresholds on the daily rainfall data.

Figure 2.16 shows that discarding the rainfall beyond x inches showed significant improvement from non-preprocessing. Discarding any rain beyond 0.1 inches showed the most improvement compared with higher thresholds. It makes sense that rainfall impacts the plant quality up to a certain level and beyond that, the water might be off before the soil could absorb it [57].

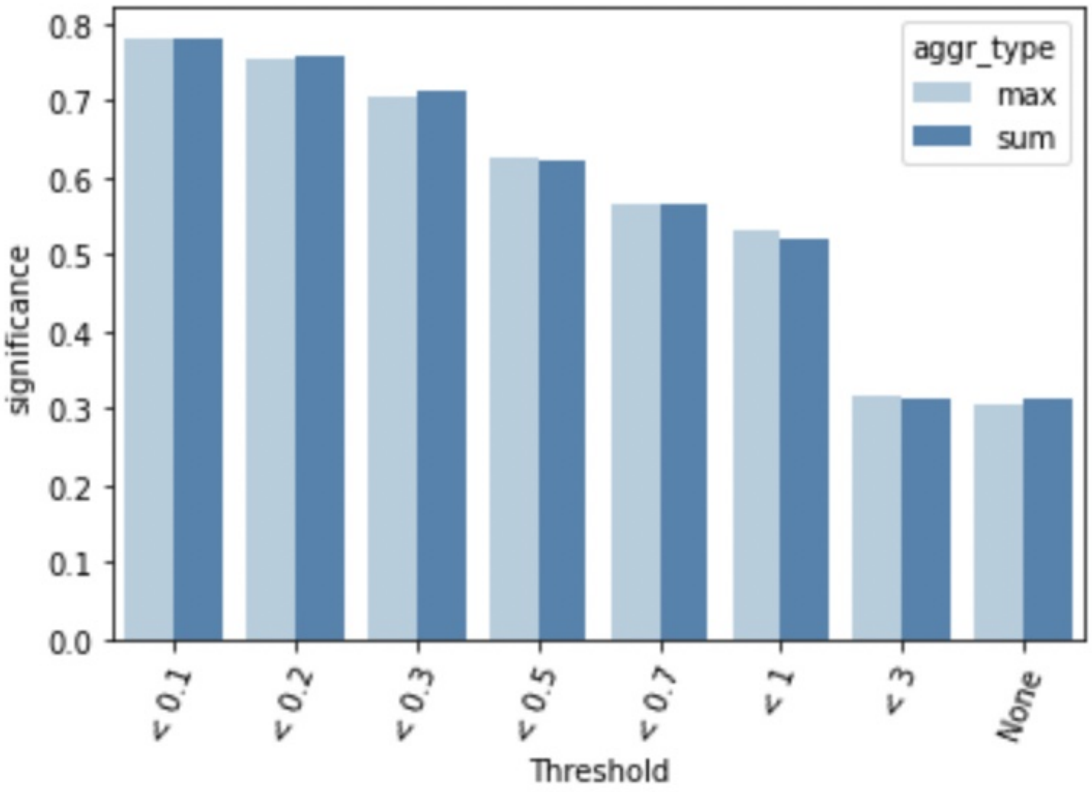


Figure 2.16. Accuracy depending on various preprocessing thresholds (upper-limit) on rainfall

Figure 2.17 shows that discarding the first x inches of rainfall did not capture the associations any better than no-preprocessing on the rainfall. We can hypothesize that ignoring rainfall up to a certain level does not show any improvement in strengthening the relationship between the rainfall and plant quality. This insight is counter-intuitive to the belief the first few inches of the rainfall is soaked up by the soil.

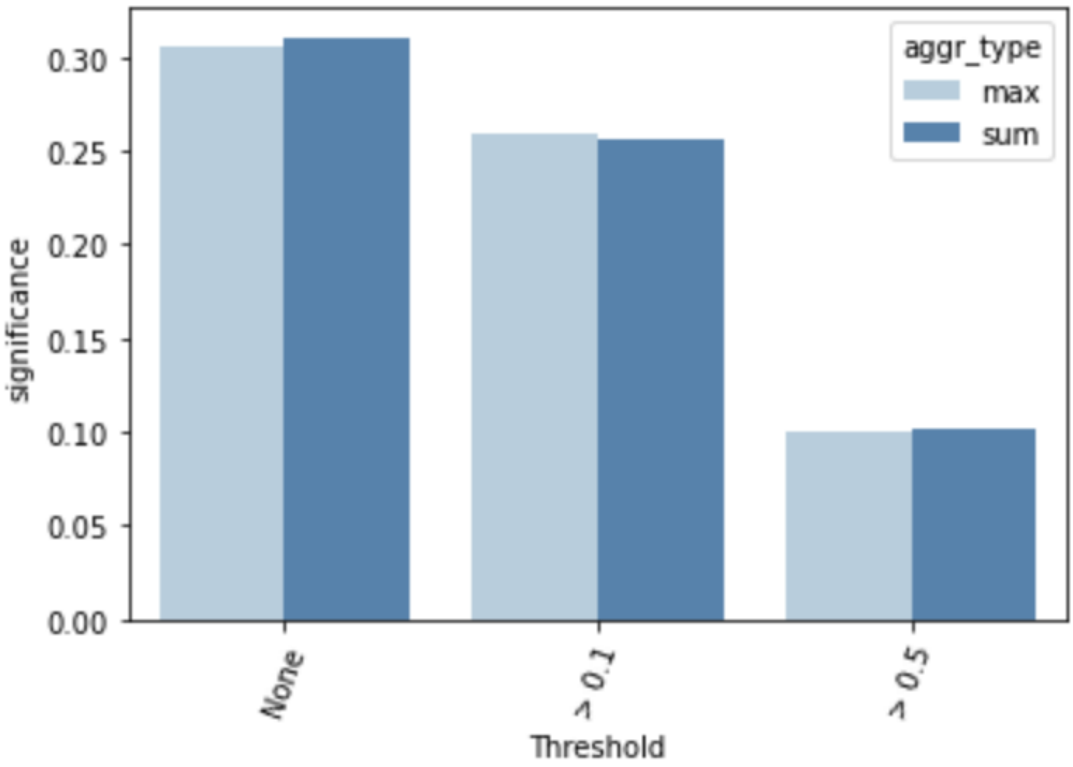


Figure 2.17. Accuracy depending on various preprocessing thresholds (lower-limit) on rainfall

2.4.5. Dependency on Sample-size

We can clearly see in Figure 2.18 that increasing the sample size drastically improves whenever we double the size of the sample set. With more and more observations, the cluster formations become more precise and eventually the association estimates become closer to the actual strength.

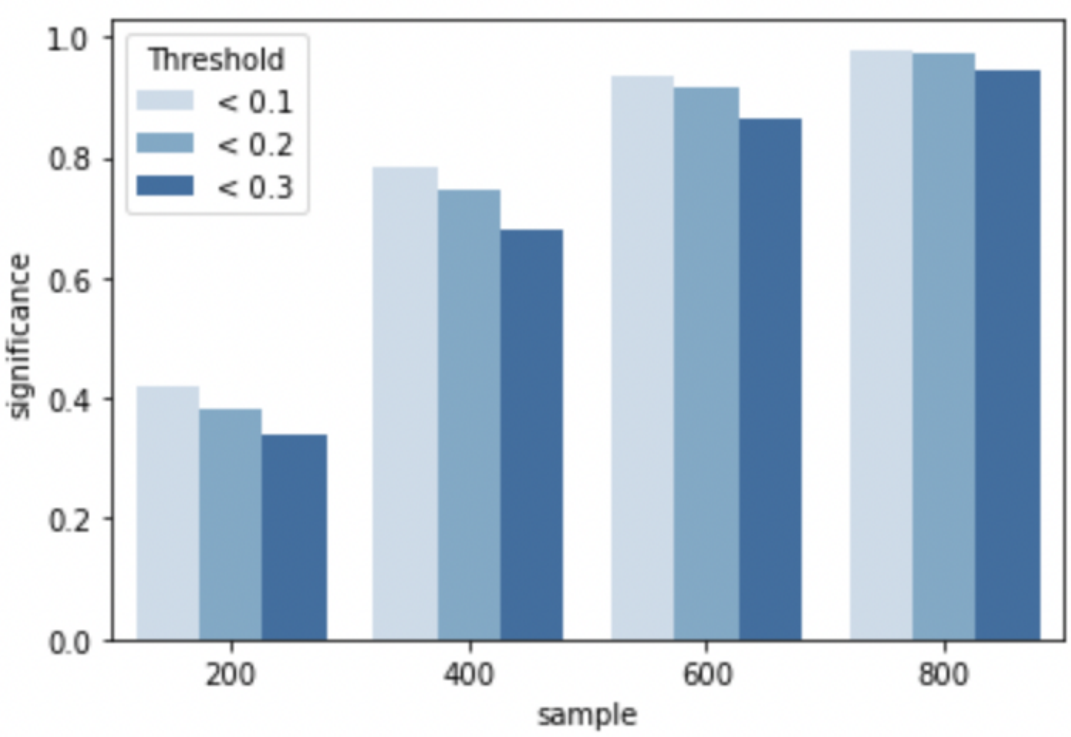


Figure 2.18. Accuracy depending on the sample size

2.4.6. Experiments on Rainfall Preprocessing for Various Intervals

Once we aggregate the experiments, Figure 2.19 indicates that weekly aggregation captures the highest association. Figure 2.19 shows the results for a sample size of 400. We chose this sample size for a trade-off between resilience and effectiveness. Figure 2.18 show that it is easy to capture the relationship between attributes with larger sample. Hence the difference between the effectiveness of the algorithm for various aggregation tends to shrink. However, too small a sample size will fail to provide enough information for the model to build upon and it will lead to overfitting.

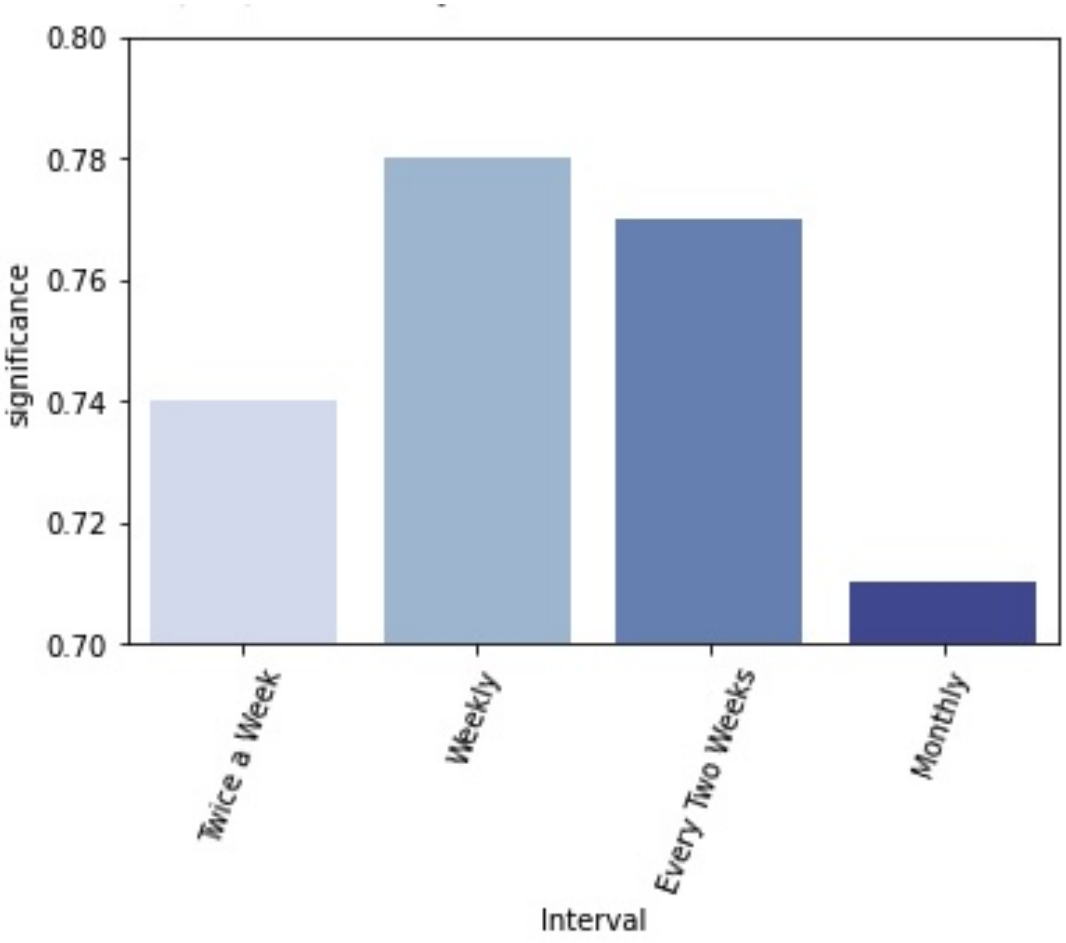


Figure 2.19. Accuracy depending on the rainfall aggregation intervals

2.5. Summary

To conclude, we have highlighted the potential challenges for big data analysis, specifically within the scope of environmental sustainability. Additionally, we demonstrated the effectiveness of quantifying the relationship between environment and agriculture attributes for precision agriculture. We proposed two algorithms to capture the non-linear relationship between the datasets without depending on the function approximation strategy. We have compared the efficiency and effectiveness of this this approach with that of the state-of-art methods for evaluating how strongly two sets of vectors are related using time series data. Additionally, we have established the process of determining various preprocessing techniques to extract insights that help to build decision management tools in precision agriculture.

3. RECOVERY ALGORITHM TO CORRECT SILENT DATA CORRUPTION OF SYNAPTIC STORAGE IN CONVOLUTIONAL NEURAL NETWORKS

Pre-trained CNN on embedded devices is vastly affected by Silent Data Corruption (SDC). SDC refers to undetected data corruption that causes errors in data without any indication that the data is incorrect and thus goes undetected. In this chapter, we propose a software-based approach to recover the corrupted bits on a pre-trained CNN due to SDC. Our approach uses a rule-mining algorithm and we conduct experimental studies on the propagation of error through the topology of the CNN in order to detect the association of the bits for the weights of the pre-trained CNN. This approach increases the robustness of safety-critical embedded vision applications in volatile conditions. A proof of concept has been conducted for a combination of a certain CNN and a vision data set. We have successfully established the effectiveness of this approach for a very high level of SDC. The proposed approach can further be extended to other networks and data sets.

The Artificial Intelligence (AI) problem space has become highly dependent on Machine Learning (ML) algorithms. AI is an interdisciplinary scientific field that is used by computer systems to build predictive models. One subfield trains a computer system to find patterns from historical data. One of the most popular branches of machine learning is Artificial Neural network (ANN), which has become the state-of-the-art in many AI application areas. ANN is based upon the idea that a machine can learn patterns from data in a similar fashion to how a human brain performs that task [58]. The concept of ANN was first introduced in 1943. The strength of ANN comes at the price of high computational complexity. That is the reason why ANN did not get much attention until the mid-80s.

Convolutional Neural Network (CNN), which is a specific type of ANN, is mostly used for computer-vision related problems. Computer vision is a branch of Artificial Intelligence that is concerned with the automation of tasks related to human visual systems. CNN is one of the most resource-intensive ANN due to its complex topology and size. However, with the rise of Graphics Processing Units (GPU) and distributed computing, CNN started to receive attention applied to

computer-vision related problems. In 2012, the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), AlexNet (a specific type of CNN), established CNN to be the state-of-the-art method for image data.

Table 3.1. Complexity of different CNN Architectures

CNN	**Year**	**Parameters**
AlexNet	2012%	60M%
Clarify	2013%	65M%
OverFeat	2013%	70M%
VGG	2014%	135M%

High-performance clusters with GPU acceleration can take care of the training phase of a neural network [59], which consumes the largest share of the computation. However, even pre-trained neural networks demand high memory due to the large number of connections in each layer (input, intermediate results, and output) in the network. The 8-layer AlexNet needs 240 MB to store 61 million weights [60] in a 32-bit floating-point format, and it is ever-growing. Table 3.1 shows how the parameter size for different CNN architectures has grown over time. For example, the number of weights in AlexNet that was introduced in 2012 was 60 million; whereas VGG (Visual Geometry Group), introduced in 2014, has 135 million weights.

Embedded systems adopted computer vision using CNN in no time since the semiconductor industry has evolved drastically over the years. Real-time embedded systems have many applications of advanced Decision Support System (DSS). A few of the good example applications of such DSS are smart environments, fitness monitoring, and military missions [61]. We will narrow down our discussion to computer vision applications in embedded systems (embedded vision). Thus, a good example of such a system is the biometric authentication system implemented in the iPhone [62]. The theoretical background of this problem is facial recognition, which is a well-known computer vision application.

However, due to the need to accommodate a massive number of weights, even the testing phase of such networks becomes a bottleneck when it comes to embedded devices because of the memory space restrictions [63]. An embedded device is dedicated to a specific task on mechanical or electrical systems [64], and are often challenged by real-time computing constraints such as inefficient power management [65]. Specialized architectures such as conv-SRAM (CSRAM) [66] has been developed to accommodate CNN in microcontrollers, which is the heart of embedded systems.

Among many real-time constraints, embedded systems suffer from data corruption due to reduced size and power supply [63]. Often, embedded systems have to operate in harsh environments, which makes them more prone to errors. For example, rugged embedded systems are designed especially for harsh environments [63].

Soft error is the predominant challenge in the semiconductor industry. [67] With the rise in the use of real-time embedded systems, off-the-shelf embedded processors have received a lot of attention lately. These systems are cursed with having to work in harsh environments. That leads to external radiation of ionizing particles terrestrial neutron, which is the major source of soft errors [68]. That, in turn, aggravates the problem of inconsistent supply voltage [69], which has been a challenge for embedded systems ever since. [70] When the charge disturbance crosses the fault-tolerance threshold of the hardware, the data state of the memory cell flips and an undetected error occurs.

There are two types of Soft Errors, Single Event Upset (SEU), and Burst errors. The scope of our experiments is limited to SEU. Burst errors involve errors in multiple bits simultaneously. Error detection becomes much more complex for this type of error. Fortunately burst errors occur rarely [71]. We can detect SEU with the help of checksum or parity bits. however, the exact position of the error remains unknown, which makes it unrecoverable. SEU very often cause Silent Data Corruption (SDC), which is very prevalent in embedded systems and thus is the prime focus of discussion in this chapter.

During the testing phase, CNNs calculate the dot product of the input for each layer and the corresponding pre-trained weights to generate the output of that layer, which is then propagated to the next layer and used as input. SDC can deviate the result drastically from the expected, which

might lead to detrimental consequences in systems since embedded vision is used in many critical real-time decision support systems such as self-driving cars [72].

There has been significant research on making embedded systems hardware resilient to data corruption. However, most of the robust hardware has a higher power consumption. For example, 8T and 6T are two varieties of SRAM (Static Random-Access Memory), and the 8T SRAM cell is a more reliable storage cell than 6T SRAM at the cost of a 40% higher power consumption. SRAM is used for the cache memory in embedded devices [73].

3.1. Related Work

Contemporary work on minimizing the impact of soft errors in embedded-vision applications can be broadly classified into two types; the first type, which comprises most of the work in the area, tries to address the problem by making error-resilient hardware using efficient power management and storage. The second type is focused on building a software-centric approach to recover the corrupted bits to reduce the overall impact. Our strategy is based on the latter category.

The Embedded Vision Engine (EVE) [74] is a specialized architecture aimed at implementing automotive vision applications (computer vision applied to Advanced Driver Assistance Systems (ADAS)). EVE provides a fully programmable architecture that is 4-12 times faster and 4-5 times more energy-efficient.

Intelligent Protection Against Silent (IPAS) output corruption [75] is an instruction replication procedure to reduce the impact of Silent Output Corruption (SOC) errors by protecting only vulnerable instructions. Numerous hardware evolved to mitigate the need for an embedded vision application. An SRAM-embedded convolution architecture [66] is designed to overcome the shortcoming of the architecture designed for generic embedded machine learning classifiers. It was tested on LeNet-5 CNN trained on the MNIST data set. It uses voltage averaging on a 266×62 Conv-SRAM (CSRAM) array.

Even with the most sophisticated hardware, embedded systems experience a substantial amount of silent errors [76]. A software-enabled approach to recover the corrupted bits could be beneficial for that. A generic [77] [78] first step of that approach is to analyze and identify the fault propagation through the application. The majority of the work on this type consists of duplication of instructions [79] to bring robustness against the silent errors, which is not feasible for the CNN based application due to the size. To give an example of an application-oriented approach, Augusto

[80] proposed a hierarchical approach to tackle uncertainties of the operational environment of Unmanned Aerial Vehicles (UAV). The gap between the frame rate of the on-board camera and the observed frame rate on-ground is utilized to build a resilient “UAV swarm".

Algorithm-Based Fault Tolerance (ABFT) [81] is a kind of a software-based approach where algorithm-specific techniques are leveraged to detect and repair silent errors. The foundation of this approach is the fact that not all silent errors have a significant impact on the performance of the application. Being able to classify the sector where the silent errors cross the tolerance threshold (impact error bound) narrows down the problem space and reduces the numeric calculation for recovery. Another set of ABFTs focuses on more generic numeric operations, such as matrix operations. Some of them focus on the checksum and roll-back recovery methods.

An exciting work that has been done in the area is a hybrid approach of a hardware and software-based technique [77]. The method leverages a software approach to identify the sensitive bits and then uses a hardware approach (selective latch hardening) to guard those bits selectively. However, with increased uncertainty in the environment of embedded applications, it became more and more challenging to improve the hardware to guarantee the performance of the embedded vision application. A pure software-based recovery approach can suffice to serve that purpose.

In this chapter, we investigate the sensitivity of SDC through SEU on the memory bits storing the pre-trained weights of a CNN, type of layers (fully connected and convolutional), and the order of them. Once we identify the sensitive bits, we propose an algorithm to find the association between those bits and to recover the corrupted bits of the memory cells based on the generated rules of the association. We will show the experimental results for the different error levels and recovery rates. There are various techniques to detect the errors that occur in a RAM, and we will use the method called checksum, which is very commonly used for the same. Checksum is derived from a block of data in order to detect errors that occurred or have been introduced during its storage.

3.2. Approach and Methodology

Association Rule Mining (ARM) [82] has been introduced in 1993. Since then, it has attracted considerable attention and has been applied in particular to market basket analysis for which customers’ buying patterns are discovered from retail sales transactions. ARM is one of the key techniques to detect and extract useful information from data. ARM’s aim is to identify strong

rules discovered in the data by using some measures of "interestingness" [83]. In particular, to select interesting rules from the set of all possible rules, constraints on various measures of significance and interest are used; the best-known measures are minimum thresholds on support and confidence.

Unfortunately, the task of ARM is computationally expensive, in particular with large data set sizes as well as when large numbers of patterns exist. Thus, we have employed the FP-Growth Algorithm that was proposed by Han [84]. The FP-Growth algorithm is an efficient and scalable method for mining the complete set of frequent patterns by pattern fragment growth. The algorithm uses an extended prefix-tree structure for storing compressed and crucial information about the frequent patterns (named frequent-pattern tree (FP-tree)).

Figure 3.1 shows the overall process of building a recovery model, which starts with training a certain CNN on a certain image data set. Although the proposed model is generic to any CNN, it must be built on the weights of a pre-trained CNN to generate the association rules, which are the backbone of this model.

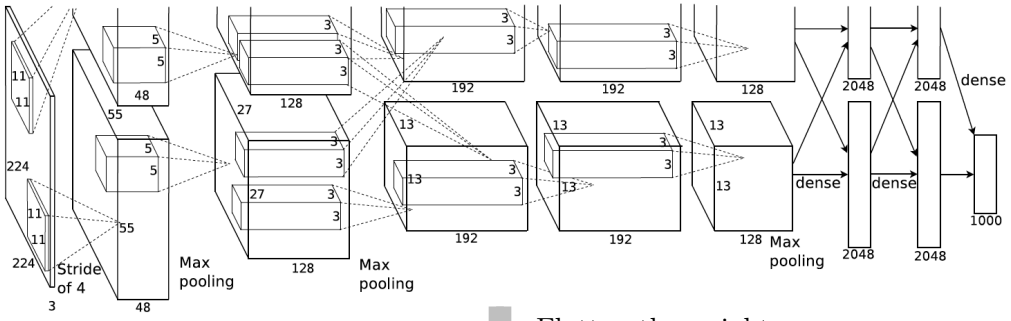
First, we have trained the Alexnet [85] on the CIFAR10 image data set [86], which yields a set of tensors. The tensors maintain the topology of the network; hence, those are flattened to combine all the weights. These weights can be denoted as (W_1, W_2, ..., W_n). Thereafter, each weight W_i is represented as a series of thirty-two bits (B_0, B_1, ..., B_3). Hence, a matrix W is created where each row signifies the binary representation of the float value of the corresponding weights. A cell from the matrix W can be denoted as W_{ij}, which represents the j^{th} bit of the i^{th} weight.

First, we detect the bits that have the most impact on the accuracy of the CNN as MSB (Most Significant Bit) (msb0-m). Then we come up with deduction logic to prioritize the prediction. In our example, the 30^{th} and 29^{th} bits always contain certain values without any exception. In case of the prediction of the corrupted bits, we prioritize these bits to check the observed values. We use a stochastic approach to build the rest of the recovery model, which is to find the association of the MSBs within these weights. We have used the FP-Growth algorithm in [87] to generate the association rules. The binary bit representation is considered as an individual item for this item set mining problem. Once we have the rules (for a certain min-confidence and min-frequency), we can build the recovery function. For a corrupted weight, each of these bits will be considered individually to calculate the likelihood of being corrupted. We have assumed that there is only

one corrupted bit for each corrupted weight. For msbi, we find the rules that have msbi=0 in the consequent and the rest of the MSBs in the antecedent and calculate the sum of confidences of those rules. Similarly, we calculate the sum of confidences for msbi=1. If the bit value (0/1) with a higher likelihood does not match with the observed value for that weight, then it is considered to be the corrupted bit. We further check if there are multiple MSB that have this mismatch. For such scenarios, the ratio of the likelihood will be considered to predict the corrupted bit. The likelihood of a bit to be corrupted is calculated as the inverse ratio of the likelihood of the observed value of that bit.

Please note that if there are two bits that have the likelihood of corruption greater than zero, then that indicates that the observed value in that bit has a lower likelihood of occurrence. Hence, we chose the bit with a higher likelihood of corruption. In case of the tie (which is fairly rare), we select randomly.

Figure 3.1 shows the process flowchart for implementing this algorithm for other Convolutional networks.



Flatten the weights into a list and portray the bit representation

	B_{31}	B_{30}	B_{29}	...	B_1	B_0
W_1	0	0	1	...	0	1
W_2	0	0	0	...	0	0
W_3	0	1	0	...	0	0
...	...	...	...	...	...	...
...	...	...	...	...	...	...
W_{n-1}	1	0	1	...	0	0
W_n	0	0	1	...	0	1

Filter the consecutive bits having the most impact on the performance of CNN

	B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0
W_1	0	1	1	0	1	1	0	1
W_2	0	0	0	1	0	1	0	0
W_1	0	1	0	1	0	1	0	0
...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...
W_{n-1}	1	0	1	0	1	1	0	0
W_n	0	0	1	1	0	1	0	1

Transform the bit values into transaction format

	$\boldsymbol{i_7}$	$\boldsymbol{i_6}$	...	$\boldsymbol{i_1}$	$\boldsymbol{i_0}$
$\boldsymbol{t_1}$	i_{71}	i_{61}	...	i_{11}	i_{01}
$\boldsymbol{t_2}$	i_{72}	i_{62}	...	i_{12}	i_{02}
...	...	...	...	...	...
$\boldsymbol{t_{n-1}}$	$i_{7\,n-1}$	$i_{6\,n-1}$	...	$i_{1\,n-1}$	$i_{0\,n-1}$
$\boldsymbol{t_n}$	i_{7n}	i_{6n}	...	i_{1n}	i_{0n}

Figure 3.1. Flowchart of Methodology

Perform Association Rule Mining to create rules

antecedent	consequent	support	confidence
$i_0 = 1$, $i_7 = 0$, $i_6 = 1$	$i_1 = 1$	0.6	0.9
...	...	...	...
$i_6 = 0$, $i_7 = 0$	$i_1 = 0$	0.7	0.85

Corrupt one of the significant bits of W_i

	B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0
W_1	0	1	1	0	0	1	0	1

Utilize the generated rules to create the likehood of corruption at each bit

	B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0
Likehood of 0 in the bit (A)	0.85	0.50	0.60	0.40	0.50	0.40	0.70	0.20
Likehood of 1 in the bit (B)	0.70	0.92	0.58	0.30	0.70	0.50	0.60	0.70
Likehood of corruption	0.82	0.54	1.03	0.75	**1.40**	0.80	0.86	0.29

likehood of corruption = a/b if the observed value for the bit is 1 else B/A

Predict the bit to be corrupted that has the highest likehood of corruption

B_3

Figure 3.1. Flowchart of Methodology (continued)

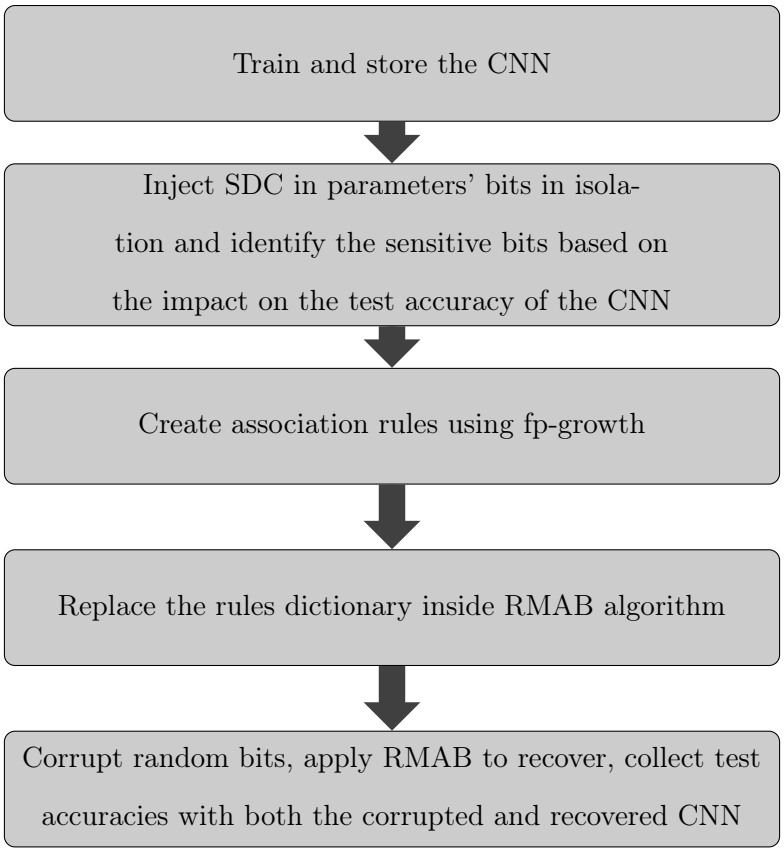


Figure 3.2. Process flowchart to implement the algorithm into other domains

3.3. Experimental Setup and Results

We have performed various experiments on a pre-trained AlexNet to evaluate the effects of soft errors on the classification result. We first investigated how the soft errors propagate through the layers, and then narrowed down the potential area for recovery and applied the recovery model.

CNN stands out from the rest of the neural networks because various types of layers come together to build the network. Nevertheless, each layer has different shapes and sizes, even within the same type. That makes it inevitable to investigate further and look deeper into the architecture of the network.

3.3.1. Structure of CNN (Alexnet)

We have used Alexnet for our experiments. Alexnet consists of five convolutional, five max-pooling, and one fully connected layer, which gives a total of thirteen layers, including the input and output layers. A pre-trained Alexnet has twelve sets of weight matrices; each set connects two

consecutive layers. The following is the structure of an AlexNet: (64, 3, 11, 11), (64,), (192, 64, 5, 5), (192,) , (384, 192, 3, 3), (384,), (256, 384, 3, 3), (256,), (256, 256, 3, 3), (256,), (10, 256), (10,).

Each line of the above structure signifies a weight matrix. The last weight matrix, indicated by (10,) is to connect the last hidden layer to the output layer. The last hidden layer is fully connected, which is the decision layer. We have trained this model on the CIFAR10 data set, which has ten image classes. For obvious reasons, we have ten weights in the final set to connect ten nodes (each of them calculating the probability of the corresponding class for a specific input). (10, 256) indicates the weight matrix to connect the last Max-Pooling layer to the fully-connected layer.

The first weight matrix, indicated by (64, 3, 11, 11), is to connect the input to the first hidden layer, which is a convolutional layer. We have used 64 filters in this layer of size (11×11). Each of these filters slides over the input matrix and collect the RGB values separately. The second index of this (first) weight matrix is always three, as the images have depth three since we have three color channels.

The weight vectors followed by every 4-dimensional matrix are used to connect the convolutional layers to the following max-pooling layers. That explains why the first dimension of the 4-dimensional matrix always matches the length of the following vector. The rest of the 4-dimensional weight matrices are to connect the max-pooling layers to the next convolutional layers.

The following hyper-parameters were used for the experiments and training of the CNN:

- Number of Epochs = 164
- Learning Rate = 0.1
- Schedule = 81, 122
- Gamma = 0.1
- Optimizer = Stochastic Gradient Descent
- Momentum = 0.9
- Weight Decay = 5e-4

The accuracy of the trained CNN model is 77.22%.

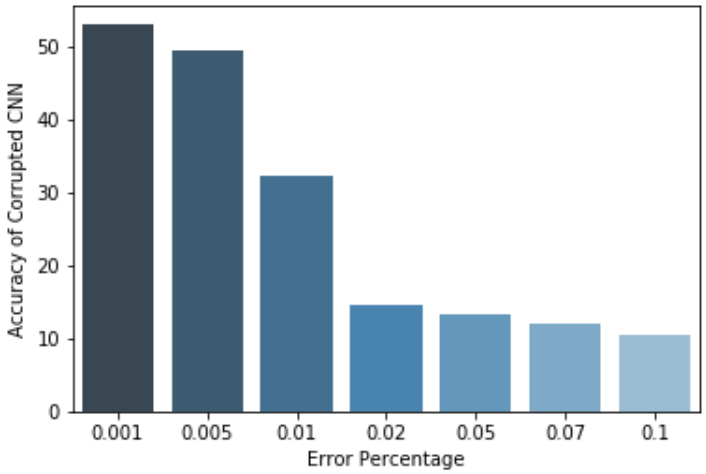


Figure 3.3. Accuracy of CNN after corruption vs. percentage of error

3.3.2. Sensitivity of CNN to Soft Errors

Figure 3.3 displays the accuracy of the Alexnet in the y-axis and the percentage of the soft error on the x-axis. The percentage of error is calculated based on the total number of weights in the network. Each weight is chosen randomly, and one of the 32 bits is randomly selected for each weight. We can see a sharp drop in the accuracy with the percentage of the soft error increasing until it reaches 0.05 and then it saturates. Please note that the accuracy of the trained model is 77.22%. In the later sections, we will observe that this behavior changes when we isolate either a layer or a bit index in the experiments.

3.3.3. Sensitivity of Layers to Soft Errors

Due to various layer types and sizes in the CNN, we have performed a layer level investigation. The layers vary significantly in size, which leads to a difference in impact on the accuracy. The larger a layer is, the more probable it becomes to be corrupted, which leads to a stronger influence on the accuracy.

Figure 3.4 shows the size (number of weights it contains) of each layer. We can clearly see that only four of the layers dominate the volume of the weights, and all of those are convolutional layers. For further investigation of the layers, we will continue with the convolutional layers because of their sizes.

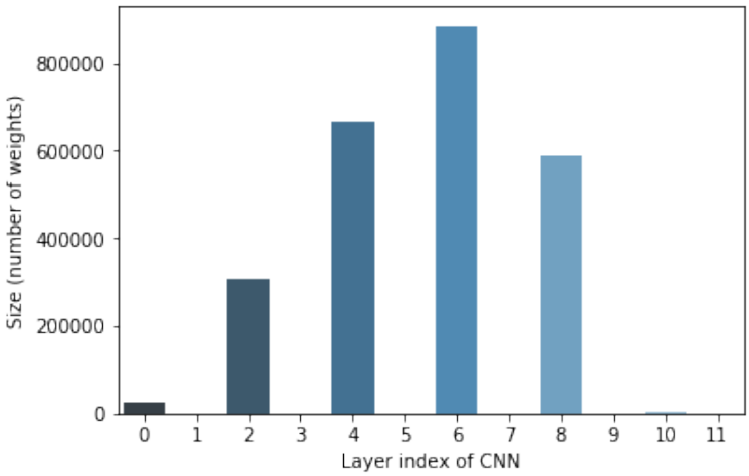


Figure 3.4. Number of weights within the layer vs. layer index

First, we examine the effect of the error on the classification accuracy when we fix a layer for the error injection. As part of this experiment, we have used the amount of error as the percentage of the layer size. That ensures to maintain a fixed likelihood of error occurrence for each layer while letting the error be generated for each layer in isolation.

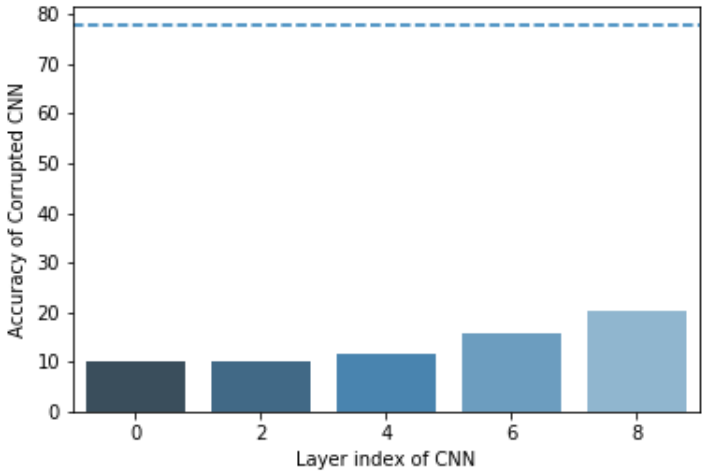


Figure 3.5. Accuracy of CNN after corruption (isolated to each layer) vs. layer index

Interestingly, the accuracy of the model does not depend on the size of the layer, as shown in Figure 3.5. Instead, we observe a clear pattern of increasing accuracy (or decreasing impact)

towards the final layer of the neural network. This indicates that the impact of the level of features (on the classification accuracy) created at every convolutional layer decreases with the index. Thus, lower level features (for example, connections) formed at the earlier stage of the CNN are more critical than higher-level features (for example, shapes or objects) created in the later stages. For the experiments shown in Figure 3.5, we have used a random selection of bits. We have repeated the same experiment by isolating the bits to inject SDC at each layer and then averaged the accuracy over all the bits; the results are displayed in Figure 3.6. In the figure, we can clearly see that the accuracy of the CNN is not impacted at all when we isolate the bits. We will investigate individual bits next to identify the reason behind this behavior.

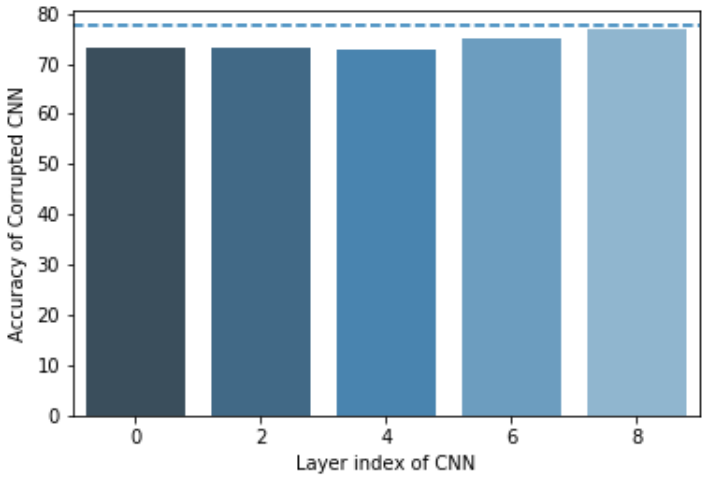


Figure 3.6. Accuracy of CNN after corruption (isolated to each layer and bit combination) vs. layer index

3.3.4. Sensitivity of Bits to SDC

Our recovery model leverages the relationship of bit positions on the float value. We will first investigate the bit level dependency for the impact of SDC on the classification accuracy. This will help us to narrow down the problem in size. The recovery models dependent on selective hardware can also benefit from this. First, we will plot the frequency of 0/1 for each bit. From Figure 3.7 we observe that the 30^{th} bit (bit index starts from 0) always contains 0 and the 29^{th} bit always

contains 1. Bits 3, 4, and 5 contain a significantly higher number of 1s compared to 0s. We will revisit this when looking at the layer-wise investigation.

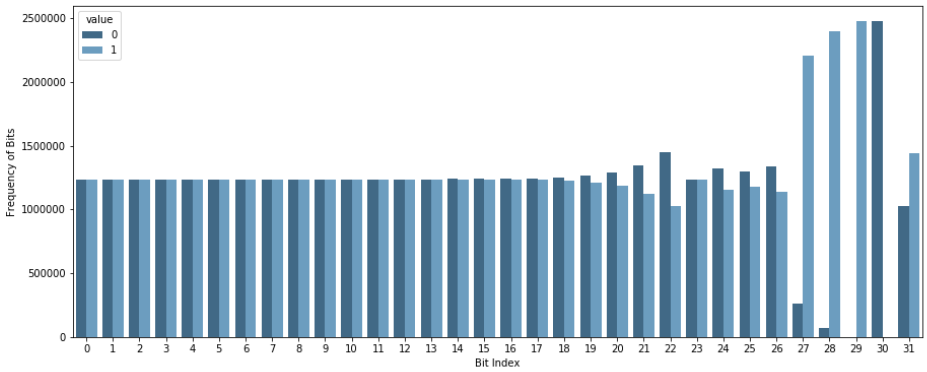


Figure 3.7. Binary value frequency vs. bit index

After that, we will investigate the impact of SDC on the classification accuracy for each bit. For that, we inject SDC on a specific bit of randomly chosen weights and calculate the classification accuracy. However, we will isolate each layer while injecting the SDC and then average the accuracy value later.

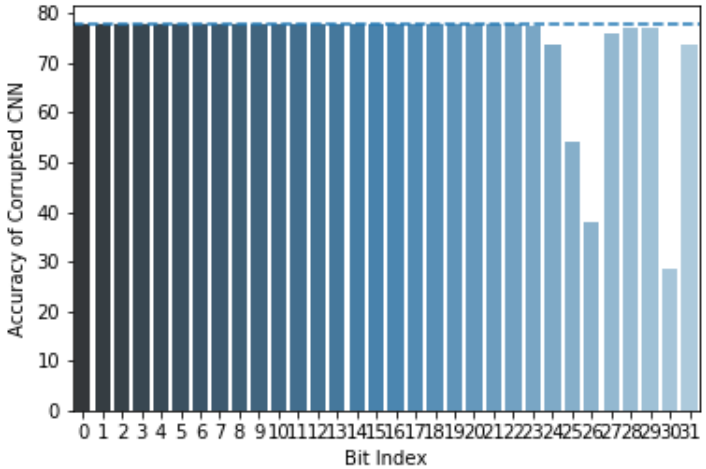


Figure 3.8. Accuracy of CNN after corruption (random layer) vs. bit index

In Figure 3.8, we can clearly see that only a few bits are sensitive to the error. We will repeat our experiment for each with a randomly chosen layer (without isolating the layer). We did not find any pattern there itself. To investigate further, we chose random weights and random bits from the weight for each iteration for injecting the SDC at various levels of error volume.

Next, we will narrow down the area to be recovered for any kind of corruption. Only certain bits will be monitored using a checksum, which will be recovered. In Figure 3.8, we see that before bit 25 there is not much impact of SDC on the accuracy of CNN. We performed a controlled experiment on the accuracy of CNN with SDC injected on the last 7 bits ($25^{th} - 31^{st}$). In each iteration, the bit and the weight were chosen randomly.

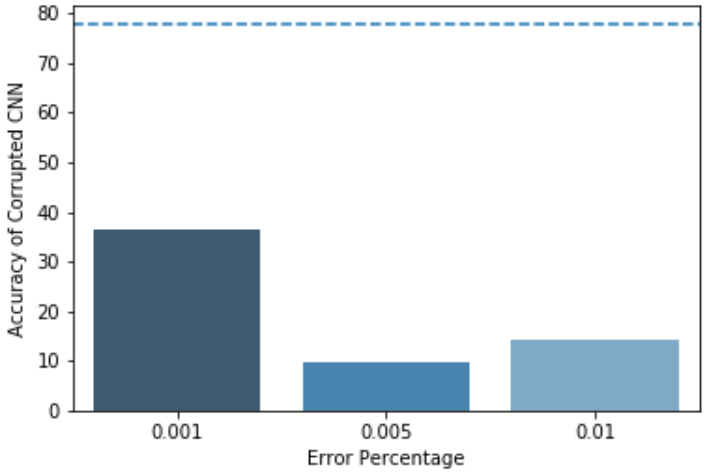


Figure 3.9. Accuracy of CNN after corruption (isolated to randomly chosen MSB) vs. percentage of error

Figures 3.9 and 3.10 show that the impact of SDC is limited to certain bits. Figure 3.9 showed the accuracy of the CNN when the SDC were restricted to the MSBs (25-31), and Figure 3.10 is for the LSBs (0-24). We can focus on protecting only the first 7 bits since the accuracy will not be affected by the SDC occurring on other bits. However, none of the experiments show any significant correlation with the percentage of error or the amount of SDC injected. The runtime cost of ARM increases exponentially with the number of bits considered. If we can narrow down the bits having a significant impact, the runtime cost of RMAB reduces drastically.

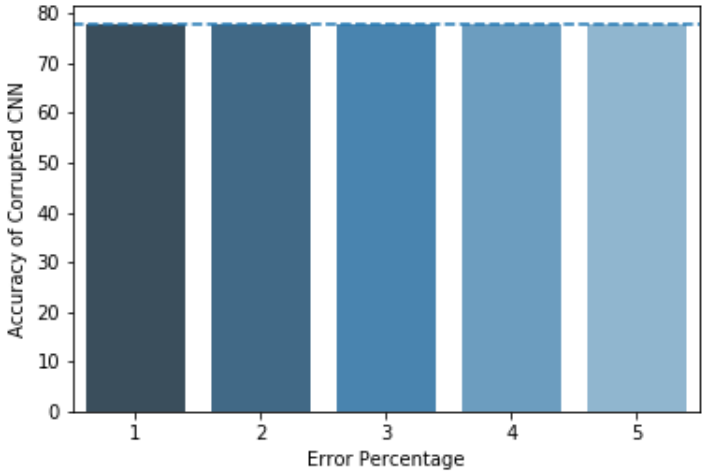


Figure 3.10. Accuracy of CNN after corruption (isolated to randomly chosen LSB) vs. percentage of error

3.3.5. Performance of Recovery Function on Significant Bits

We maintained a checksum bit for Bit 0-7 to detect any occurrence of SDC. Once an error is detected on these bits, the 29^{th} and 30^{th} bit values are checked as these two bits always have certain values. If Bit 29 and Bit 30 do not have the expected values in the corrupted weight, the recovery algorithm is applied to predict the bit that is most likely to have been corrupted with the combination of the other bits.

First, we test the performance of the recovery model when the SDCs were injected on the MSBs being isolated. The results are shown in Figure 3.11. Bit 29 and Bit 30 are removed from the figure since those are recovered using deductive logic. However, those were considered for error injection. We see that the model does not perform so well when the SDC is injected on Bit 31. However, when we inject the error in randomly chosen bits (Bits 25 to 28) as shown in Figure 3.12, we achieve high performance after recovery. The performance of the corrupted model received a lot of impacts even for such a low error rate. Although the accuracy of the corrupted model does not show a clear trend, the intention of the experiment was to confirm the high accuracy of the recovered model.

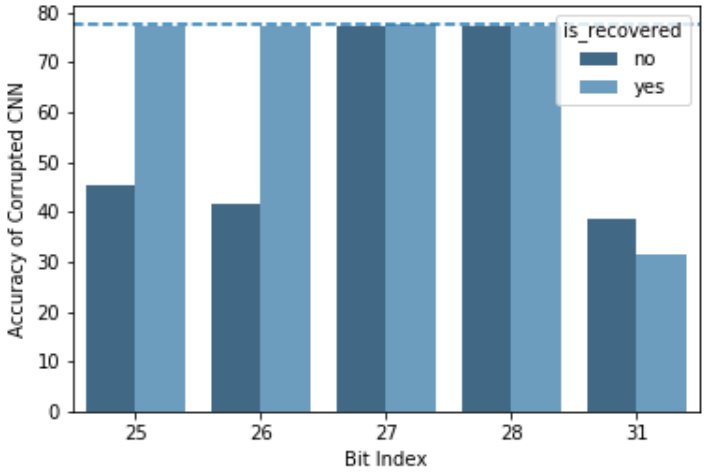


Figure 3.11. Accuracy of CNN after corruption (random layer) with or without recovery vs. bit index (only MSB)

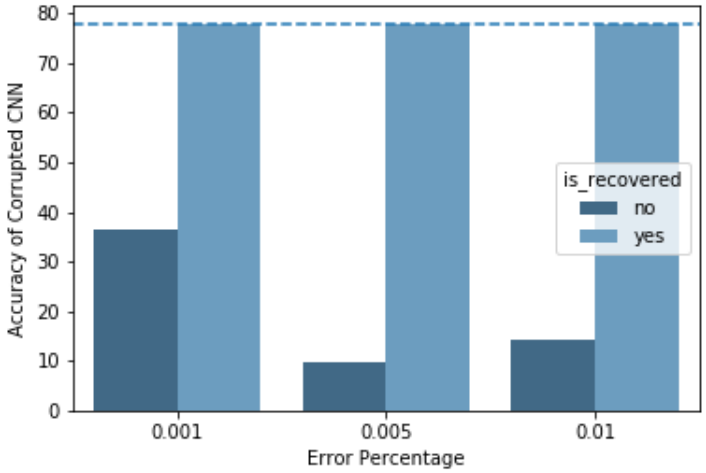


Figure 3.12. Accuracy of CNN after corruption (isolated to randomly chosen MSB) with or without recovery vs. percentage of error

3.3.6. Performance of Recovery Function on Convolutional Layers

When we repeat the experiments of recovery for isolated layers, we see that the accuracy values are successfully reached compared to the original model irrespective of the layer (refer to Figure 3.13).

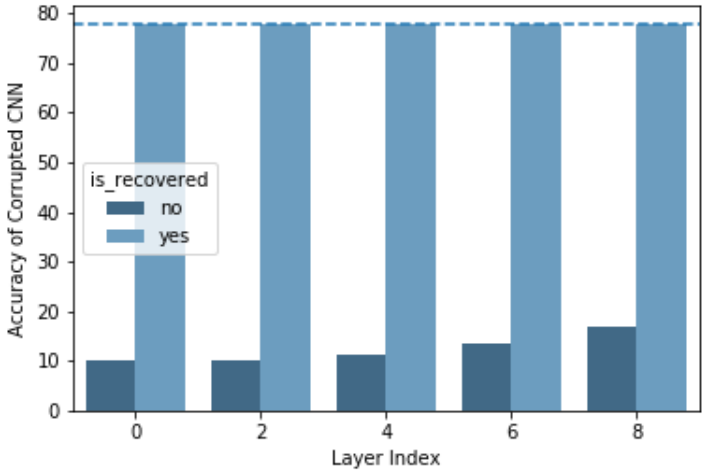


Figure 3.13. Accuracy of CNN after corruption (isolated to each layer) vs. layer index

3.3.7. Holistic Performance of Recovery Function

The real evaluation of the recovery model can be tested with the random selection of bits and weights, where the accuracy of the model was impacted the most. In Figure 3.14, we can see that the performance slightly decreases (monotonously) with an increasing amount of SDCs. However, even with a 20% corruption rate (which is enormous in an embedded system environment), the recovery performance is outstanding.

The result of each experiment is averaged over thirty iterations to test the stability of the experimental results. The standard deviation of the accuracy through the iterations are plotted in Figure 3.15. The low standard deviation values of the accuracy for the experiments confirm the stability of the recovery model. Even though the recovery model did not perform well when the SDCs were injected on Bit 31 isolated, it performed well when the bits were randomly selected, which was the major weak area for the corrupted models in embedded systems.

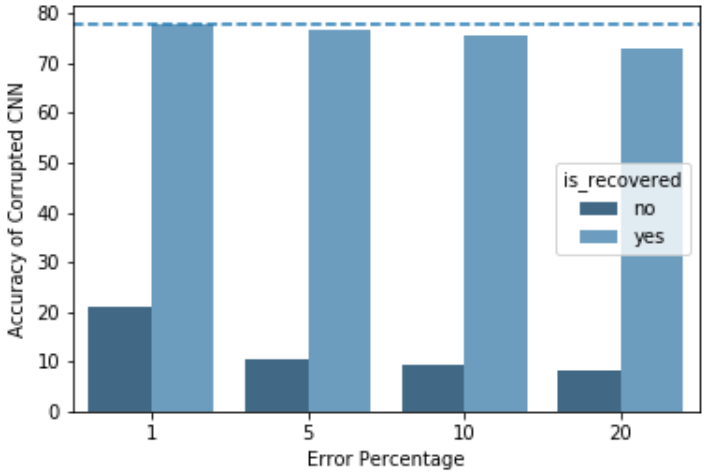


Figure 3.14. Accuracy of CNN after corruption with or without recovery vs. percentage of error

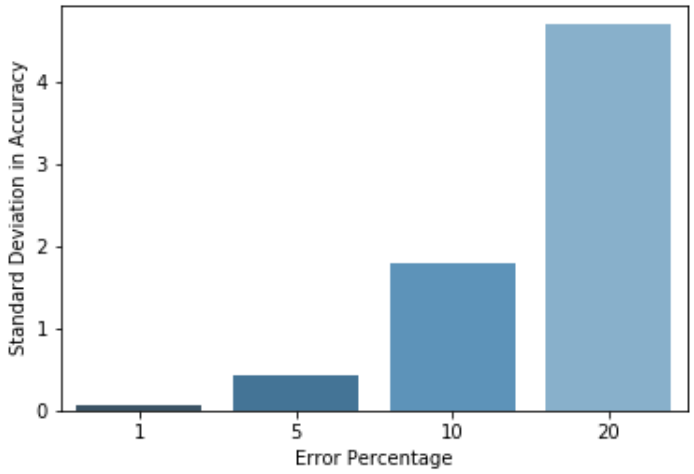


Figure 3.15. Standard deviation accuracy values of CNN for multiple experimental iterations vs. percentage of error

3.4. Summary

This chapter addressed the problem of Silent Data Corruption (SDC) on a pre-trained CNN in embedded systems. The proposed approach employed an association rule mining algorithm to correct data corruption. We performed controlled experiments on the propagation of errors through the topology of the AlexNet trained on the CIFAR10 data set. Further investigations were conducted to identify the sensitive segments of data based on its impact on model performance. Thus, we converted the weights of the pre-trained network into a set of items (binary strings) and generated rules to detect the association among the bit values. These rules are then further used to build a recovery system. For each bit, the likelihood of containing zero or one, depending on the state of the remaining bits, were calculated. Finally, we performed controlled experiments on the recovery algorithm to demonstrate the performance with various experiment parameters.

The experiments establish the effectiveness of the recovery algorithm at various levels of data corruption. We have observed that the impact of SDC is much higher on the performance of the model when we target multiple bits for corruption on various weights (single bit for each weight, though). The proposed method successfully recovers the errors, and thus, the performance of the model does not get affected much even with high levels of corruption.

As for future work, we can extend this work to other CNN models such as VGG, FaceNet, and with more massive image data sets such as CIFAR100. With each combination of CNN and data set, a new pre-trained model will be prepared for data corruption, then a sensitive section of the bit level storage must be identified, and the recovery model needs to be rebuilt based on the new rules of associations among the sensitive bits.

4. GENRE BASED HYBRID FILTERING FOR MOVIE RECOMMENDATION ENGINE

With the dramatic rise of internet users in the last decade, there has been a massive rise in the number of daily web searches. This leads to a plethora of data available online, which is growing by the days. A recommendation engine leverages this massive amount of data by finding patterns of user behavior. Movie recommendation for users is one of the most prevalent implementations. Although it goes way back in the history of recommendation engines, collaborative filtering is still the most predominant method when it comes to the underlying technique implemented in recommendation engines. The main reasons behind that are its simplicity and flexibility. However, collaborative filtering has always suffered from the Cold-Start problem. When a new movie enters the rating platform, we do not have any user interaction for the movie. The foundation of collaborative filtering is based on the user-movie rating. In this paper, we have proposed a hybrid filtering to combat this problem using the genre labeled for a new movie. The proposed algorithm utilizes the nonlinear similarities among various movie genres and predicts the rating of a user for the new movie with the associated genres for the movie.

As the internet keeps accommodating more and content for users, the relevance of the content for users has become crucial. On the one hand, more content expands the options for users. On the other hand, it increases the complexity of finding the relevant information for users. In consequence, the task of recommendation engines has become more and more difficult. The recommendation systems available today can be broadly classified into three major categories; collaborative-filtering, content-based-filtering, and hybrid of the first two[88][89][90] [91].

Collaborative-filtering (CF) is grounded on identifying similar (nearest neighbors) items or users for a new movie-user pair and estimate the rating based on a certain aggregation method. For item-based CF, similarities among the movie are calculated based on the ratings by mutual users. For user-based CF, similarities among the users are calculated based on their ratings of mutual movies watched. These similarity matrices are used to identify the nearest neighbors (movies or users).

Content-based-filtering has always been important in the field of recommendation engines because it addresses the Cold-Start problem [92]. Both user-user CF and item-item CF are dependent on the user-item interaction. When a new movie is introduced to the recommendation engine, it does not have any interaction data with the user at that point. Without existing ratings by the users, it is not possible to find similar movies, which is the foundation of item-item CF. Also, as no user has rated the movie, there is no way we can recommend this movie to new users that are similar to the ones that have already liked this movie. This is the foundation of user-user CF. Additionally, data sparsity is predominant in any movie review data set. Movies with an insignificant number of user ratings show inferior performance compared to the ones with a significant number of user ratings[93][94].

However, content-based CF finds similar movies based on the metadata of the movie such as director, length of the movie, language, etc. [95] proposes a content-based filtering technique where attributes are extracted from the ratings, and this deals with data sparsity. Although it handles cold start problems to some extent, it entirely depends on the content of the movies. The disadvantages of this approach are twofold, operations cost, and quality of recommendation. From the standpoint of operations cost, the collection of content of movies is expensive as those are hand-engineered and requires a lot of domain knowledge. With the increase in the number of movies across the globe, it is becoming nearly impossible to generate features for those. Also, this process of feature extraction is counter-intuitive. Furthermore, the quality of the recommendation is entirely dependent on the quality of the feature generation process. Its ability is limited in terms of expanding users' existing preferences.

Collaborative-filtering calculates the similarities among user ratings to make predictions. This method is only purposeful when users have rated a significant number of mutual items. Additionally, the absence of content of the movies makes it impossible to match similar users unless the same items were rated by them. For example, suppose userA liked the movie "Lord of the Rings", and userB liked the movie "Lord of the Rings II". Although there is a similarity in their preferences, collaborative-filtering will fail to match them. A hybrid technique of collaboration with content tackles this shortcoming by combining information captured by both content-based-filtering and by collaborative-filtering. In collaboration with the content-user, the similarity matrix is built using

both the content of the movies as well as the rating of movies in common. The feature extraction based on the content of the movie is done by content-based techniques.

The weights of the extracted features designate the importance of the corresponding feature to the user. Like collaborative-filtering, the similarities between users are calculated. However, the weights of the features are used in place of user ratings to calculate the similarity. This approach has lots of features to calculate the similarities between users. Hence, it deals with the issue of users having an insufficient number of rated movies. Moreover, the extracted features are capable of capturing users' preferences outside of their usual environment. This is a considerable uplift from content-based-filtering. However, we still need a significant number of users that have rated a movie for the recommendation. Similar to collaborative-filtering, it is not possible to recommend a brand-new movie unless there is a user who has rated it.

The rest of the paper is organized in the following way. Section 4.1 discusses the limitation of the state-of-the-art methods on cold-start scenarios and how this paper contributes to mitigate those issues. Section 4.2.1 explains the algorithm with mathematical notations and 4.2.2 shows the flowchart diagrams. Section 4.3.2 describes the process of data preparation to simulate the cold-start scenario. Section 4.3.3 and 4.3.4 shows the experimental results on the MovieLens-100k data set with various levels of data sparsity. Section 4.3.5 shows the experimental results on the MovieLens-25m data set. In addition, we have further improved the hybrid filtering with nonlinear multivariate similarity metrics. The comparative experimental results are shown in Section 4.3.6.

4.1. Related Work

Various methods have been proposed over time to build product recommendation engines. Due to the variety of available data and the use cases, there is no one-size-fits-all technique. However, most of the proposed methods can be broadly classified into two categories, content based filtering and collaborative filtering. Content based filtering also includes context based filtering. They use time and space features of the items and users [96]. Collaborative filtering, in contrast, leverage the interaction among the items and the users to predict the rating of a user-movie pair. Furthermore, collaborative filtering methods are broadly classified into two categories, nearest neighbor based models [97] and matrix factorization based models [98]. Nearest neighbor based filtering methods focus on the similar users or items to estimate the likely rating. Matrix factorization is captures the latent relationships between the users and the items, project that to a low-dimensional space

and compute the similarity in the reduced space [99]. The most common matrix factorization is Singular Value Decomposition which was also used for the winning algorithm in the Netflix Prize [100]. Although the latter category has gained popularity lately, the former category still dominates the industrial level implementations. The prime reason behind this is its simplicity and intuitiveness [101]. Due to the same reason, Nearest Neighbor based models win over more sophisticated models in a variety of domains. For example, prediction of remaining useful life on degradable products have been dependent on models based on pattern recognition models [102]. These models are grounded on statistical estimation of the decaying process. However, in many scenarios, nearest-neighbor based models win over the former mentioned sophisticated models due to its simplicity [103].

Any form of collaborative filtering suffers from groups of users known as gray and black sheep. Unusual users with inconsistent preference compared to the other users are referred as the Gray sheep [104]. Black sheep refers to the group of users that have idiosyncratic tastes [105]. The time complexities of collaborative filtering is linear and with the growing size of the user-movie rating data, it becomes difficult to keep up with the runtime of these algorithms [106]. It is very common that synonymous (or the exact same) items with different names occur in a recommender system data set. Collaborative filtering fails to detect those unless users have shown similar behavior towards rating them.

[107] has proposed a content-based filtering approach that calculates the contents' similarity. Movie genres are defined by content experts. Hence, it is more reliable than system-generated features. Additionally, it deals with the Cold-Start problem as similarities are calculated only based on content. It calculates the mutual genre count to estimate the similarities among genres. However, it does not consider the user rating of each genre while calculating the similarities. Additionally, the experiment is performed on a test set of 10 movies, which fails to establish the reliability of the proposed method.

[108] proposes a recommendation system based on the extracted low-level visual features from the trailers of the movie. The authors have used visual feature extraction as an alternative to the metadata of the movies. This method is dependent on the machine-generated features, whereas genre information is far more reliable as it is generated by domain experts. Also, the algorithm-guided feature generation is an unstable area in computer vision. Furthermore, this approach fails to expand users' existing tastes as it does not consider the interaction among the high-level components

of the movie. [109] proposes a similar approach but for videos. As the videos do not have man-made sample data (trailer for the movie), it utilizes the whole content of the video to extract the feature. This is not feasible for movies.

[110] uses the plot summaries of the movies to generate natural language processing based features. The authors combine the high-level meta-data information with the low-level features generated by the plot summaries. Similar to [108], it fails to expand users' existing preferences. Also, the bag of words approach can be totally misleading while defining the theme of the movie.

[111] calculates the relative importance of genres to each user from the review matrix. The weighted average importance of the user of the corresponding tagged genres of a movie is estimated as the predicted rating of that user-movie pair. This approach restricts the prediction of users' existing preferences as it does not consider the similarities among the genres. For example, if a user has rated 'Batman' as 5, and he has watched only one sci-fi movie, the relative importance of 'action' genre for the user becomes 5. Whereas, in reality it could be the case that he rated it high because it was an 'action' film. This model would incorrectly recommend all the sci-fi movies to the user.

[112] proposes a content-based recommendation system that leverages six high-level features, such as the origin of the movie, which is a major limitation of this approach. Most of the real-time training data sets contain selection bias. For example, if a user has mostly watched Russian movies, this model will most likely limit the predictions within Russian movies whereas, in today's movie world streaming platforms generate audio in multiple languages. Features like the genre is more universal and does not have this limitation.

[113] extracts the movie interest of users from user tweets. Although the authors aim to solve a different problem with an entirely different type of data set, we share the same foundation in terms of the approach, which is the recognition of user interest.

[114] proposes a recommendation technique for users through mobile devices. The system leverages the GPS information from the movie search operations on mobile devices. The primary pitfall of this approach is the assumption of user behavior. We should not assume the inference of user touchpoints based on intuition. If a user has searched a movie on a mobile device, then it does not necessarily show his interest if he did not watch the movie.

The method implemented in [115] leverages the generated features of the genre. It enriches the diversity of the recommendation as it utilizes genre data. This paper attempts to define genre diversity using a Binomial framework, which is different from our problem statement. However, it shows the robustness of genre data in terms of enhancing the diversity of the recommendation.

The cold start problem can also be addressed using the similarities among the users based on their interactions on social networks [116]. This approach tries to fill the gap in the rating data to tackle the data sparsity. However, it is heavily dependent on an external data source, which is extremely difficult to stitch with the rating data set. In an ideal world where all the users from various platforms are centrally connected with network governance, this method can prove to be effective. Although, that is far from feasible to apply on a real-life recommendation platform.

[117] proposes an algorithm to deal with cold start problem. They calculate the latent features with deep neural network using the users data (when available), genre information and movie title. However, there are various limitations on this method. For Movie Lens 1m data set, this algorithm is dependent on the user demographic information. User demographic information is not readily available for many new recommendation systems. Hence, this method is limited to the recommender systems that are able to capture the user demographic information. For example, MovieLens-100k does not have user information. Hence, it is limited to the users that allow their demographic information to the system. Even with the provided demographic, the reliability of the information is often questionable. Hence, we do not rely on the user demographic data. Secondly, the simulation of the cold-start data is done based on the acquisition of the user. This paper has considered the users acquired in the last year. We have simulated the data in such a way that the users have at most 20 ratings. For example, in Movie Lens 100k data set, if we consider the ratings from 2003, which is the latest year on the data set, only 878 ratings are coming from the users having less than 21 ratings. Whereas there are 3348 ratings posted in 2003. Our data preparation simulates the cold-start problem closer to this work due to higher data sparsity.

[118] proposes an interesting ensemble method where various components of the rating systems are ensemble with dynamically generated weights. This work achieves an impressive error rate in MovieLens-100k data set. However, it does not establish the method to address the problem of cold-start. The data preparation included the whole data set; there was no cold-start simulation. We have captured the latent associations among the genres and combine that with the collabora-

tive filtering and provided the proof of concept how it outperforms the standard algorithms in the simulated data set.

In a nutshell, we are proposing a hybrid filtering method where the genre information of the movies is used with the association among the users in terms of similar movies. Unlike many other contents of movies such as extracted features from the media, genre data is very easy to gather. Additionally, genre information is reliable as it is not machine-generated. This proposed algorithm will be effective in dealing with the Cold-Start problem. The major contribution of this algorithm is to expand users' preferences across genres. We have implemented the algorithm using the ML-100k data set and performed robust experiments to validate the effectiveness of the proposed method.

4.2. Approach and Methodology

4.2.1. Algorithm Description

In this section, the methodology of the proposed algorithms for predicting movie ratings based on genre information will be explained. Let us start by denoting the entities involved in this experiment, in particular we have three major components.

$\{U_1, U_2, \ldots, U_p\}$ is the set of users that contribute to the review data set. $\{M_1, M_2, \ldots, M_q\}$ is the set of movies that are utilized for rating by the set of users mentioned earlier. Each of the movies belongs to multiple genres, and $\{G_1, G_2, \ldots, G_{18}\}$ is the set of genres. The rating data set is:

$$R \in r(i, j)$$

where $r(i, j)$ denotes the rating of user U_i for M_j.

We first calculate the average rating of each user for each genre. This is calculated from the user-movie rating matrix combined with the movie-genre adjacency matrix.

4.2.1.1. Genre-Average Hybrid Filtering - GAHF

Genre-average hybrid filtering (GAHF) is applicable to the test cases where the movie has at least one genre that has been tagged by the user. For example, if (U_i, M_j) is the test case where we are trying to predict the rating of a movie denoted by M_j by the user denoted by U_i. If M_j is tagged under genres $G_{1\ldots m}$, and under the assumption that U_i has rated all the tagged genres, then the predicted rating is calculated as:

$$\sum_{k=1}^{m} r(i, k)$$

where $r(i, k)$ denotes the average ratings for movies of genre G_k by user U_i.

4.2.1.2. Genre-Similarity Hybrid Filtering - GSHF

We propose genre-similarity hybrid filtering (GSHF) that utilizes the similarities between every pair of genres and overcomes the limitation of of the test user having to rate at least one genre of the test-case. In other words, it is capable of working on the use cases where the user has never rated a genre that the movies fall under. We calculate the genre-genre similarity matrix from the user-genre average rating matrix. This provides the similarity between every pair of genres based on the average rating of every user to the genres. We have considered various similarity metrics which will be discussed in Section 4.2.3.

We will utilize the genre-genre similarity matrix to predict the rating for a user-movie pair where the user has never rated the genres of the movie. For example, if (U_i, M_j) is the test case for which we are predicting the rating of movie denoted by M_j by the user denoted by U_i. If M_j is tagged under genres $G_{1,2,...,m}$, the predicted rating is calculated as:

$$\sum_{k=1}^{m} r(i).sim(k)$$

where, $r(i)$ is the average-rank vector by U_i for the genres except the ones in G; and $sim(k)$ is the similarity vector of G_k with the corresponding genres.

4.2.2. Process Flowchart Diagrams

Figure 4.1 is a process flowchart that visually describes the data preparation for our experiments to simulate the cold-start situation. And, Figure 4.2 is a process flowchart that explains the overview of the execution of the model on a new test case.

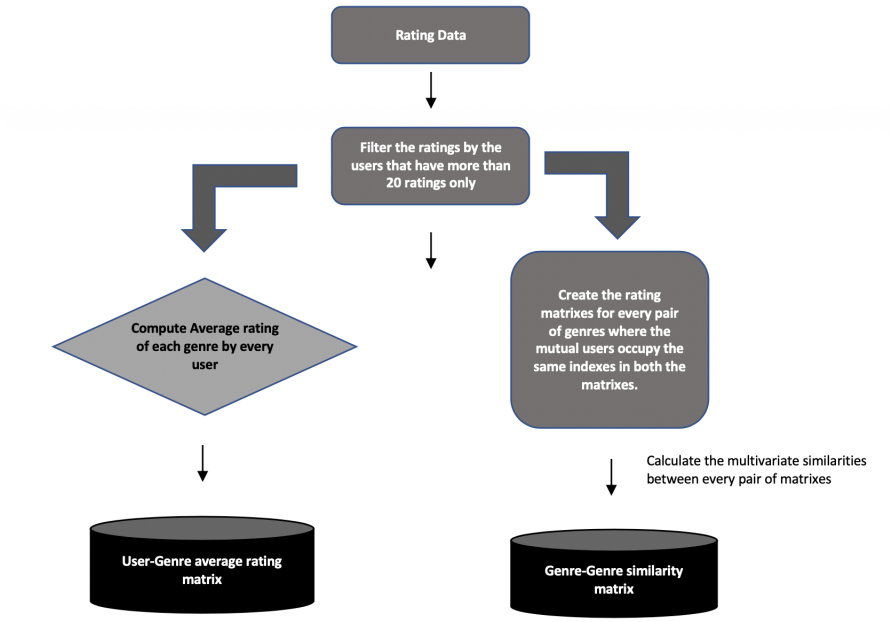


Figure 4.1. Process Flowchart for the data preparation for cold-start simulation

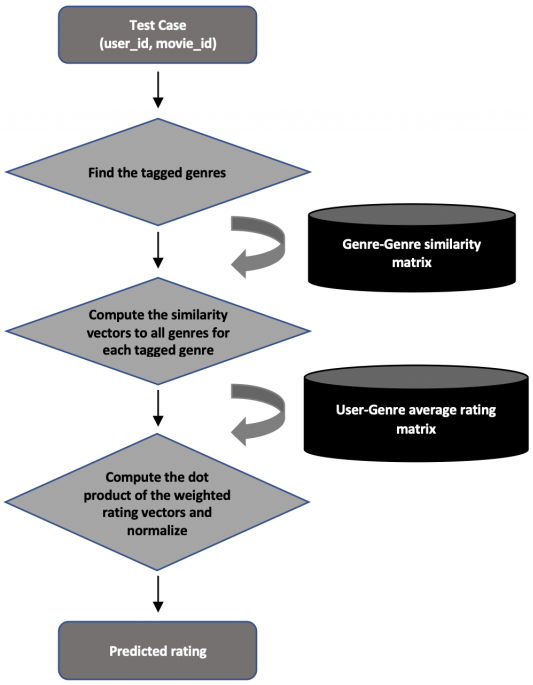


Figure 4.2. Process Flowchart for rating prediction for a test case

4.2.3. Multivariate Similarity Metrics

Canonical Correlation (CC) capture the linear relationship between two multivariate data sets. It finds the linear combinations of each set of the variables that yield the maximum correlation between them. In other words, we construct a weighted average of the variables, which is called variate. Then, we calculate the Pearson Correlation between the variates.

For example, let us suppose, $\mathbf{X}$ and $\mathbf{Y}$ are the ratings vectors by the mutual set of the users between G_i and G_j, respectively. These set of vectors are denoted as,

$\mathbf{X} = (X_1, \ldots, X_p)'$ and $\mathbf{Y} = (Y_1, \ldots, Y_q)'$

To find the Canonical Correlations, we need to construct the linear combinations:

$$Z = u_1 Y_1 + u_2 Y_2 + \cdots + u_p Y_p$$

and

$$W = v_1 X_1 + v_2 X_2 + \cdots + v_q X_q$$

such that r_{ZW} is a maximum.

To calculate Canonical Correlation, we start with the covariance matrix between $\mathbf{X}$ and $\mathbf{Y}$.

Let S be XY covariance matrix denoted as,

$$S = \begin{bmatrix} S_{yy} & S_{yx} \\ S_{xy} & S_{xx} \end{bmatrix}$$

Next, we partition the covariance matrix as below,

$$A = S_{yy}^{-1} S_{yx} S_{xx}^{-1} S_{xy}$$

Eigenvalues of A are canonical correlations squared, therefore

$$R_c = \sqrt{\mu_i^2} = \mu_i$$

As mentioned earlier, Canonical Correlations can capture the linear relationships between two multivariate data sets. There are a few non-linear algorithms that capture the nonlinear rela-

tionships. Both Nearest Neighbor Intersection (NNI) [45] and Cluster Overlap (CO) [56] work well on these tasks.

We are going to implement NNI to substitute CC in Section 4.3.6 to portray how the performance improves with this similarity metric.

NNI calculates the nearest neighbors based on the average rating vectors for each user based on both the genres and calculates the mutual nearest neighbors to estimate the similarity between them. First, we identify the of the nearest neighbors of an instance m in both data sets. Every neighbor has a probability of $k_{nn}/(N-1)$ to be in the subset defined by x_m and a probability of $k_{nn}/(N-1)$ to be in the subset that's defined by y_m. Aggregated, the expected number of mutual neighbors between any pair of random neighborhoods is $\frac{k_{nn}^2}{N-1}$.

The number of mutual observations of neighborhoods is aggregated over every possible instances m yields in a sum of

$$n_{\text{expect}} = E\left(\sum_m |U_l^x \cap U_l^y|\right) = \frac{k_{nn}^2}{N-1}$$

if column vectors $\mathbf{X}$ and $\mathbf{Y}$ are unrelated.

Post aggregation, the expected sums are big enough for a Poisson distribution to be approximated by a Gaussian distribution. We used Poisson distribution for comparison. After the approximation, we collect the z-score, which is how many standard deviations above the mean,

$$z = \frac{\left|\sum_m |U_m^a \cap U_m^b| - n_{\text{expect}}\right|}{\sqrt{n_{\text{expect}}}} \tag{4.1}$$

For the CO algorithm, we count the number of overlapping data points between every pair of clusters, each generated on different data sets. And, we compare that with the expected number of overlaps with random chance.

$$\chi^2 = \sum_{i=1}^{k_{\text{clust}}} \sum_{j=1}^{k_{\text{clust}}} \frac{\left(|C_i^X \cap C_j^Y| - \frac{|C_i^X|\,|C_j^Y|}{N}\right)^2}{\frac{|C_i^X|\,|C_j^Y|}{N}} \tag{4.2}$$

where C_i^X and C_j^Y are the members of clusters X_i and Y_j respectively, i.e. $|C_i^X \cap C_j^Y|$ is the number of observations in the overlap of the two clusters, and $\frac{|C_i^X|\,|C_j^Y|}{N}$ is its expected overlap between the two independent clusters. We can consider the χ^2 distribution to calculate the probabilities which show the strength of the relationship.

4.3. Experimental Setup and Results

4.3.1. Data Sets

We have used the MovieLens data set [119] to build a proof of concept and to evaluate our proposed approach. It has various versions of the data set which are identified by the size of the rating matrix. We have considered the most commonly used version, which is also the smallest, ML-100k. Additionally, we have used the largest version, ML-25ml, to establish the robustness of the algorithm. ML-100k has 100,000 ratings where $1,682$ movies are rated by 943 users. Each rating is an integer ranging from 1 to 5. It has 93% sparsity, which means if we create an adjacency matrix on the user-movie combinations, then 93% of the cells will be empty. ML-25ml consists of 25 million ratings as the suffix identifies. It has $162,541$ users and $59,047$ movies. The ratings range from 1 to 5, but with an interval of 0.5.

Table 4.1 shows the data based on the user id, movie id, and the rating.

Table 4.1. Movielens-100K rating data set

userId	**movieId**	**rating**
877	381	4
815	602	3
94	431	4
416	875	2
500	182	2

4.3.2. Data Preparation for Experiments

The prime motivation of this paper is to predict the rating of movies that have never (or very few ratings) been rated by the user population. This is a classic problem in the world of

recommendation engines and is referred to as the 'Cold Start' problem. We intend to utilize the genre information of the movie to address this issue. We build the training and testing data set in such a way that reflects this situation and can establish the effectiveness of our algorithm.

We have calculated the average rating of each genre for the user. For each user-movie combination, we calculate the average of average ratings of each tagged genre; it is zero if a user has not rated a certain genre. The rationale for the same comes from the 'Missing Not At Random' (MNAR), which states that the missing values indicate non-ignorable non-responses.

We have executed two sets of experiments. First, we executed experiments to establish the Genre similarity based hybrid filtering with comparative results for Memory based Collaborative filtering methods. We prepared the data in such a way that the test items appears on the train data set less than a certain frequency. We calculated the similarity matrix between every pair of genres and estimated the rating of a movie considering the average rating of each genre by the users combining the similarities among them. The second experiments expand the comparison results on the performance of the algorithm using various similarity metrics.

4.3.2.1. ML-100k Data Set

We performed the experiments on the subsets where there are limited number of ratings for each user or item. In one experiment, we prepared the data in a way that the users have rated less than a certain number of movies. In the other experiment, we prepared the data in a way that the movies have been rated by less than a certain number of users.

Furthermore, for the subset, we have split the data into train and test buckets with random sampling with a 9:1 ratio. The data set is large enough for random sampling to maintain similar distributions over various strata. For the interest of future reference, we are going to call these frequencies as per-user-rating-count and per-item-rating-count, respectively.

Table 4.2 shows the size of the training data for various per-user-rating counts and per-item-rating counts.

Table 4.2. Sizes of training and testing split for various frequency counts

Frequency	**Train size for per-user-rating**	**Train size for per-item-rating**
30	4257	7446
40	7299	10476
50	10376	14656

As we simulate the cold-start environment, we are purposefully introducing data sparsity, which results in test cases where the user has no rating in the train split. We have dropped such cases from the test case. If it is a movie that has never been rated in the train split, we can utilize the genre similarities to come up with a prediction, which is however not the case for a new user. Also, collaborative filtering algorithms fail to predict the rating if there is no mutual users or items. For those cases, we have used the overall average rating of the movie.

Table 4.3 shows the percentage of data for various ratings on both the training and the testing splits. This split is shown for the subset where the users have less than 50 ratings.

Table 4.3. Percentage share on data for various ratings on training and testing split for per-user-rating-count 50

Rating	**Training split**	**Testing split**
1	5.8%	6.1%
2	10.9%	9.4%
3	24.3%	26.8%
4	34.6%	36.3%
5	24.3%	22.4%

Table 4.4 shows the percentage of data for various ratings on both the training and the testing splits. This split is shown for the subset where the items have less than 50 ratings. For the interest of future reference, we are going to call this frequency as per-item-rating-count.

Table 4.4. Percentage of various ratings on training and testing split for per-item-rating-count 50

Rating	**Training split**	**Testing split**
1	14.6%	14.6%
2	17.4%	18.4%
3	31.1%	29.1%
4	25.3%	25.3%
5	11.6%	11.4%

4.3.2.2. ML-25ml Data Set

We have performed similar experiments on a subset of the ML-25m data set. We have filtered in the users that have 20 ratings in the full data set. From $4,611$ such users, we have randomly picked 500 for our experiments. Table 4.5 shows the percentage of the data for each rating value before the random sampling, training split after random sampling, and the testing split. We can see that the distribution is maintained after the random sampling.

Table 4.5. Percentage of ratings on training and testing split for 25m data set

Rating	Before random Sampling	Training split	Testing split
0.5	3.2%	2.9%	4.5%
1	4.2%	4.2%	4.0%
1.5	2.6%	2.4%	3.2%
2	6.8%	6.9%	5.6%
2.5	4.6%	4.6%	3.5%
3	19.1%	18.8%	18.1%
3.5	8.3%	8.6%	8.3%
4	23.6%	24.5%	24.0%
4.5	7.8%	7.5%	7.5%
5	19.8%	19.9%	20.8%

4.3.3. Experiments on Per-User-Rating-Counts

4.3.3.1. Comparative Performance

To measure the performance on the test split, the Mean Absolute Error (MAE) and the Mean Squared Error (MSE) are calculated. The results are collected with three decimal places. We have compared the performance of the proposed algorithm with both Item-Item Collaborative Filtering (IIHF) and User-User Collaborative Filtering (UUCF). Furthermore, as we are working with genre information, we have also reported the performance of Genre-Average Hybrid Filtering (GAHF).

Figure 4.3 and Figure 4.4 show the comparative performance of the Item-Item Collaborative Filtering (IIHF), User-User Collaborative Filtering (UUCF), Genre-Average Hybrid Filtering (GAHF), and Genre-Similarity Hybrid Filtering (GSHF) for various settings of the data sets.

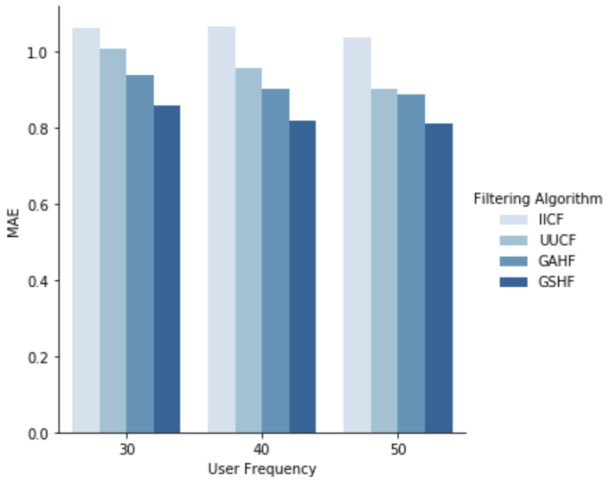


Figure 4.3. MAE for various per-user-rating counts

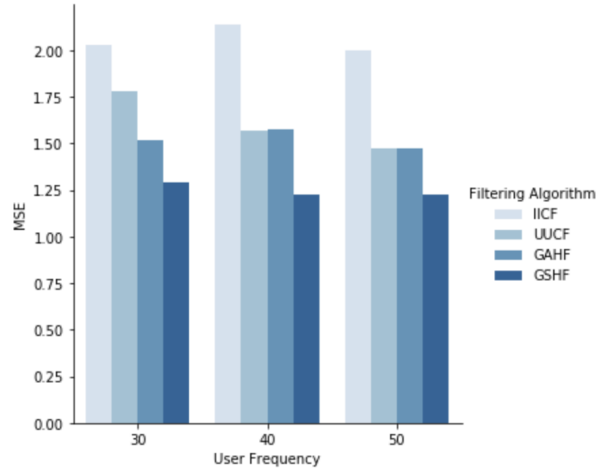


Figure 4.4. MSE for various per-user-rating count

We observe a downward trend in loss for UUCF as the per-user-ratings-count increases. As the per-user-ratings-count increases, there are more mutual users for a test-movie to leverage which leads to lower loss. GSHF, on the other hand, maintain a steady performance across the per-user-ratings-count. It establishes that GSHF is not impacted by the sparsity of the data as much, which makes it a perfect choice for the movie recommendations under the cold-start scenario.

Additionally, we observe the loss of each of the algorithms are much higher than the reported loss on the ML-100k data set by the state-of-the-art algorithms. This is, for obvious reasons, due

to the cold-start simulation. We are going to continue with the data preparation for the cold-start simulation in this paper.

4.3.3.2. Genre Count

Table 4.6 shows how the prediction loss of GSHF varies over the genre counts. We observe that the prediction losses are stable across the various genre counts. It proves the robustness of the performance of the algorithm. Although GSHF leverages the genre information, it holds equal power while dealing with the cases with small number of tagged genres. The similarities of the tagged genres with other genres suffice the information.

Table 4.6. Prediction loss for various genre counts of movies

genre count	**MAE**	**MSE**
1	0.845	1.318
2	0.803	1.177
3	0.749	1.046
4	0.882	1.48
5	0.5	0.5

We observe interesting values for MAE and MSE for genre count 5. Once investigated further, there were only 4 such test cases and the predictions are shown in Table 4.7.

Table 4.7. Predictions for the cases with 5 tagged genres

Actual rating	**Predicted rating**
3	4.0
5	4.0
4	4.0
3	3.0

4.3.3.3. Confusion Matrix

Figure 4.5 shows how the prediction values are distributed around the actual ratings. The actual labels are given on the y-axis whereas the predicted values are given on the x-axis. The first diagonal contains the correct predictions. Based on the figure we can see that most of the ratings revolve around rating class 3 and 4. Also note that a rating of 1.0 was never predicted.

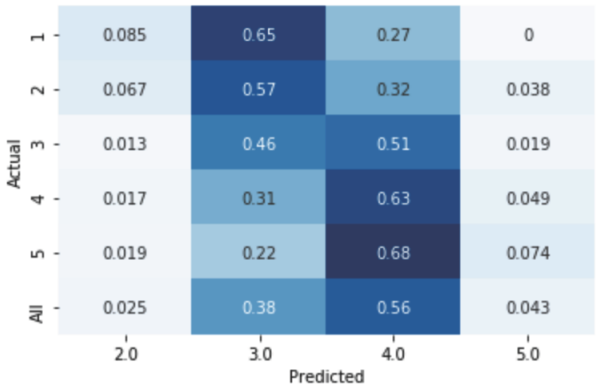


Figure 4.5. Heatmap of the confusion matrix

4.3.4. Experiments on Per-Item-Rating-Counts

4.3.4.1. Comparative Performance

For the next experiments we prepared the data in a way that the items have received less than a certain number of ratings. Figure 4.6 and Figure 4.7 show the comparative performance of the algorithms for the data preparation with various per-item-rating counts.

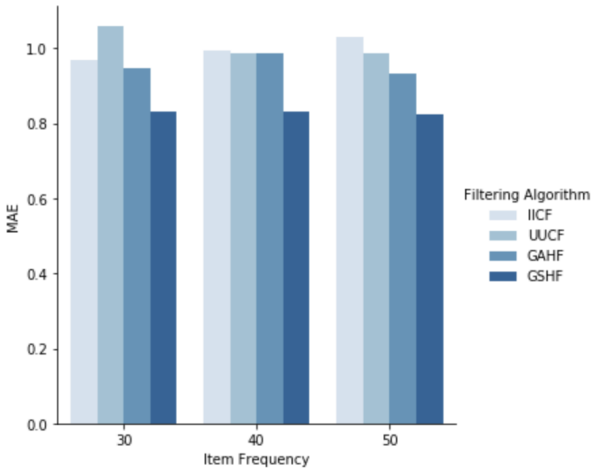


Figure 4.6. MAE for various per-item-rating counts

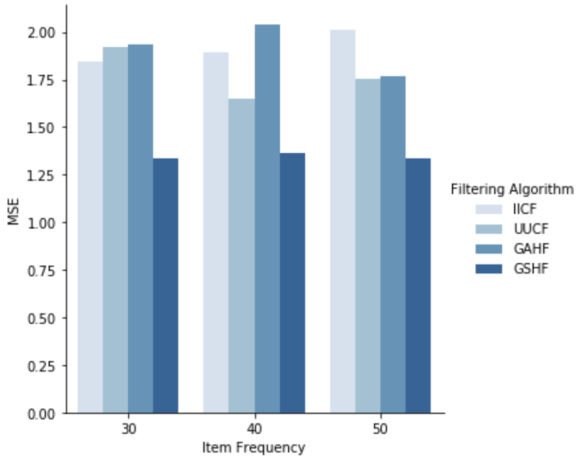


Figure 4.7. MSE for various per-item-rating count

Unlike Figure 4.3 and Figure 4.4, Figure 4.6 and Figure 4.7 do not show any trend on the performance of UUHF or IIHF with per-item-ratings-count. However, GSHF shows stability and superiority in performance.

4.3.4.2. Genre Count

Table 4.8 shows how the prediction loss of GSHF varies over the genre counts.

Table 4.8. Prediction loss various genre counts of movies

genre count	MAE	MSE
1	0.809	1.298
2	0.826	1.329
3	0.912	1.574
4	0.895	1.421
5	1	1

Similar to the Table 4.6, MAE and MSE for genre count 5 are also interesting in Table 4.8. Once investigated further, there were only 2 such test cases and the predictions are shown in Table 4.9.

Table 4.9. Predictions for the cases with 5 tagged genres

Actual rating	Predicted rating
3	4.0
4	3.0

4.3.4.3. Confusion Matrix

Figure 4.8 shows how the prediction values are distributed around the actual ratings. The matrix indicates that most ratings predict rating value 3 as indicated by the color shading.

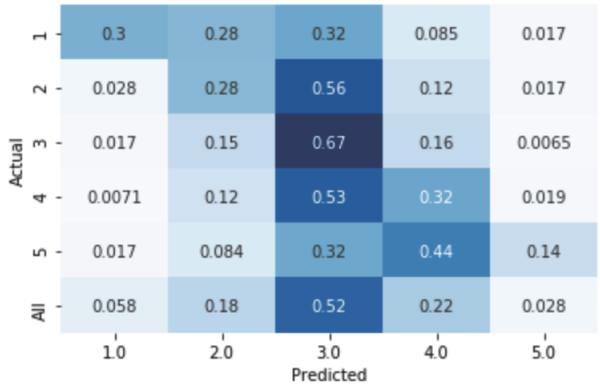


Figure 4.8. Heatmap of the confusion matrix

4.3.5. Experiments on 25ml Data Set

4.3.5.1. Comparative Performance

Table 4.10 shows the comparative performance on all 4 algorithms for the aforementioned subset of the ML-25m data set. We can clearly see that both GAHF and GSHF have smaller MAE and MSE values compared to the comparison approaches UUCF and IICF. GSHF outperforms GAHF with the lowest values of 0.792 and 1.104 for MAE and MSE, respectively.

Table 4.10. Comparative prediction loss on ML-25ml

Algorithm	**MAE**	**MSE**
UUCF	1.119	1.987
IICF	1.029	1.962
GAHF	0.847	1.266
GSHF	0.792	1.104

4.3.5.2. Genre-count

Table 4.11 shows how the prediction loss varies over the genre counts. We can observe a sharp decrease in loss for genre-count higher than 4. There are only 9 cases with genre-count 6, and 6 cases with genre-count 7. We can explain this decrease in loss as a bias. However, there are

45 cases with genre-count 5, which is large enough to eliminate the random chance of the loss to be low.

Table 4.11. Prediction loss various genre counts of movies on ML-25ml

genre count	MAE	MSE
1	0.838	1.139
2	0.754	1.096
3	0.797	1.133
4	0.835	1.169
5	0.767	0.839
6	0.611	0.472
7	0.667	0.667

4.3.5.3. Confusion Matrix

Figure 4.9 shows how the prediction values are distributed around the actual ratings. The ratings are shown in 0.5 increments. Given the color shading we can see that most of the predicted ratings range from 2.5 to 4.0.

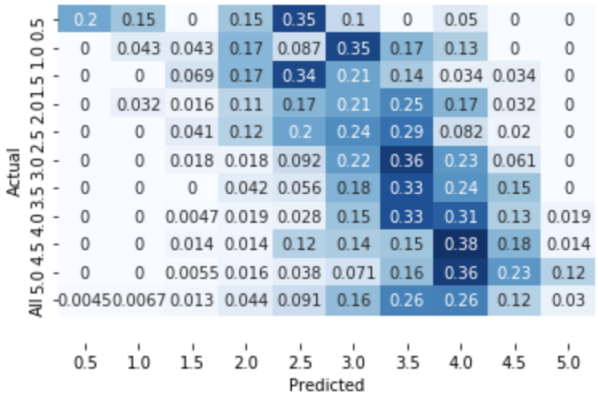


Figure 4.9. Heatmap of the confusion matrix

4.3.6. Similarity of GSHF

4.3.6.1. Experiments on ML-100k Data Set

In this section, we will compare the performance of GSHF with two different similarity metrics, namely NNI and CCA. As we discussed in Section 4.2.3, NNI is capable of capturing the non-linear relationships between two data sets, which is not the case for CCA. Figure 4.6 and Figure 4.7 show how the prediction loss varies for GSHF with CCA or NNI at various per-user-rating-count values. We observe a slightly downward trend in the loss as the per-user-rating-count increases, although there is no monotonous trend. GSHF consistently performs better (lower loss) with NNI than with CCA as the similarity metric.

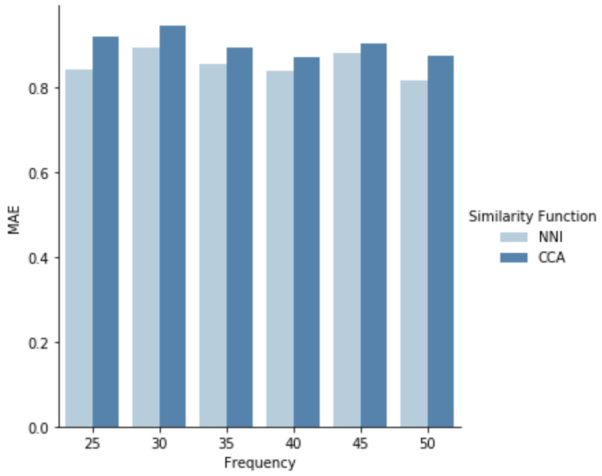


Figure 4.10. MAE for various per-user-rating counts

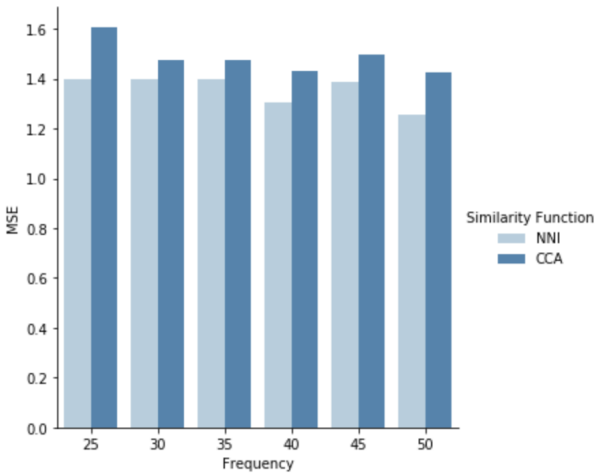


Figure 4.11. MSE for various per-user-rating-count

Figure 4.12 and Figure 4.13 show how the prediction loss varies for GSHF with CCA or NNI at various per-movie-genre-count for ML-100k. CCA performs slightly better than NNI if we consider the MAE for genre-count 4.

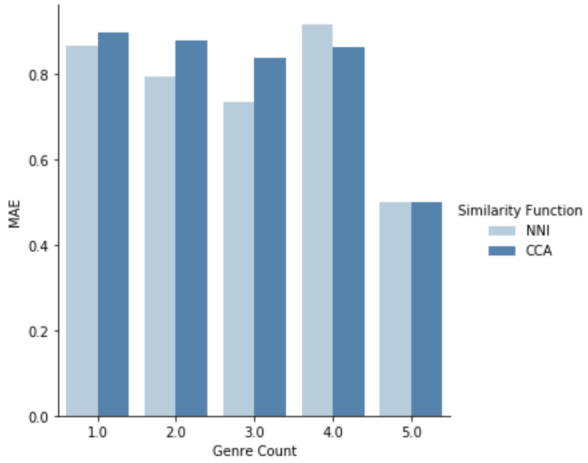


Figure 4.12. MAE for various genre counts

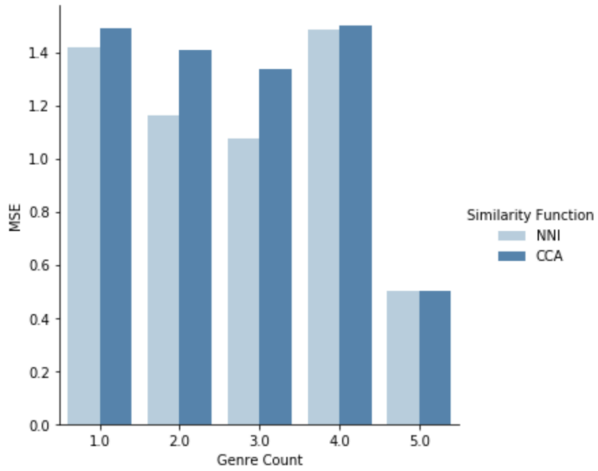


Figure 4.13. MSE for various genre counts

4.3.6.2. Experiments on ML-25ml data set

We performed similar experiments on the 25ml data set. However, for the interest of execution time, we only performed the experiment for per-user-rating-count as 50. Figure 4.14 and Figure 4.15 show how the prediction loss varies for GSHF with CCA or NNI at various per-movie-genre-count for ML-25ml.

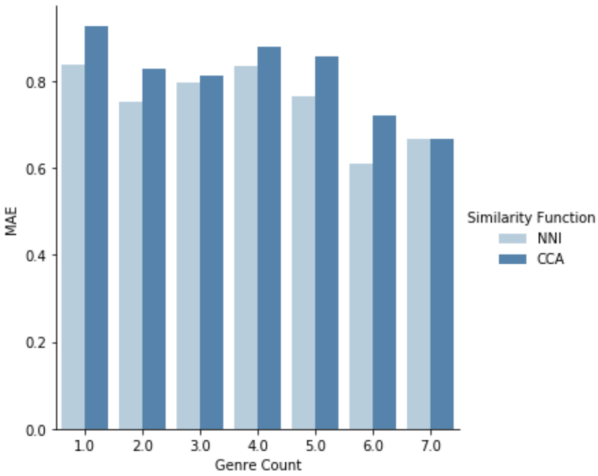


Figure 4.14. MAE for various genre counts

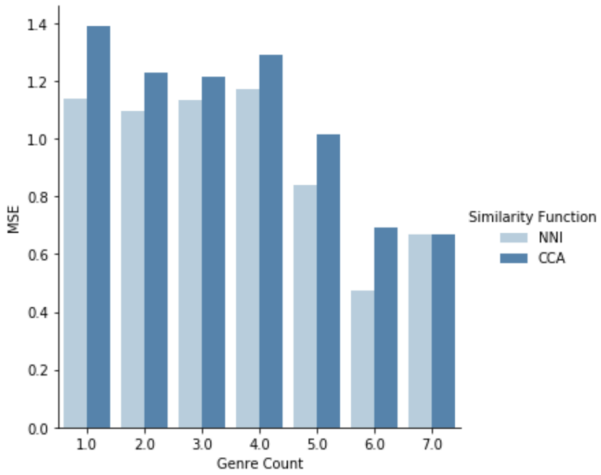


Figure 4.15. MSE for various genre counts

4.4. Summary

We have proposed a hybrid filtering technique that considers movie genres as the content of the movies. Genre is a reliable content measure on the movie as it is generated by the content experts such as directors. This algorithm is not sensitive to Cold-Start problems or limitations on users' preferences. We have applied this approach to the Movie Lens data sets for various use cases. It has consistently shown superior performance over the existing approach to establish its effectiveness. For instance, on the subset of ML-25ml data set, where each user has less than 20 ratings, GAHF and GSHF show a 18% and 23% lower MAE respectively than that of item-item collaborative filtering. Furthermore, the current existing approaches fail to expand users' current current preference. The proposed method can help an online streaming service to grow by expanding users' interests. Additionally, we have plugged in a similarity metric that is able to capture the nonlinear relationship between genres and reported the superiority in effectiveness.

As part of future work, we aim to apply our approach to various domains within the product rating platform. For example, we can implement this algorithm for online retail stores with product categories where users rate the products online. Furthermore, we can include additional reliable content on the movie or users such as demographic data to enhance the Genre based Collaborative Filtering model.

5. CONCLUSION

To summarize, this dissertation emphasizes the association among various entities within big-data-driven applications. It shows the impact in various domains; namely, environmental sustainability, embedded systems, and recommendation systems. Environmental sustainability applications can benefit from the insights generated from the systematic associations among the environmental attributes and agricultural attributes. A preprocessing technique has been demonstrated in a precision agriculture application. Embedded systems have suffered from silent data corruption which can lead to performance degradation of machine learning models deployed into them. A software-based recovery model has been proposed to address this problem using the bit-level associations within the model parameters. The impact of the model in an example scenario has been demonstrated for the same. Furthermore, the data sparsity problem is inherent in recommendation systems. A community similarity-based hybrid filtering method has been proposed for a movie recommendation system. It demonstrates how utilizing the associations among the movie genres can improve the performance of the applications. Overall, the chapters of this dissertation target various industries that heavily use pattern recognition, choose one such application for each of the industries, identify one prevalent problem for the corresponding application, define the problem statements from the stand-point of data science study, generate insights from the associations among the entities involved in the corresponding applications, and finally demonstrate the effectiveness of the proposed solutions to leverage those insights.

Chapter 2 demonstrates the potential contribution of data science in applications related to environmental science. Specifically, Precision Agriculture can benefit from the interaction between rainfall and crop properties. The experiments that are shown in the chapter are limited to a few agricultural attributes of sugar beets to establish the benefit of a couple of preprocessing techniques on the daily rainfall. However, this study can be extended to diverse crops. Which in turn leads to various crop attributes. Moreover, the proposed techniques can be applied to diverse vegetation attributes such as vegetation index derived from satellite images. The proposed algorithms are capable of identifying associations between two time series datasets as well. Additionally, there are various environmental variables other than rainfall such as wind speed, humidity, temperature, etc.

Chapter 3 illustrates the potential of software based recovery models for silent data corruption on a specific type of machine learning model. More precisely, it is designed on a very specific type of neural network. It can be extended to various machine learning models solving diverse problem statements. Due to the inherent nature of the design of the experiments, this algorithm has to be tailored to the target use case after carefully identifying the area and the scope of improvement. For example, consider a Transformer algorithm trained for Natural Language Processing tasks deployed on smart glasses. First, it will need a diagnosis on how the effect of the silent error flows through the network. Thereafter, a recovery model can be trained on the model parameters based on the diagnosis. Although the foundation of implementation will be very similar, the development process might vary drastically for each use case.

Chapter 4 addresses the problem of data sparsity in recommendation systems through interactions among the virtual communities. However, it is limited to a movie database and review ratings. This approach can be utilized on various datasets. For example, most of the recommendation systems capture the text reviews along with the numerical ratings. Virtual communities can be found from the review text using machine learning tasks like topic modeling. The interactions between these communities can extract insights through association mining. Furthermore, these insights can address the cold-start problem to some extent. The experiments shown in this chapter can be used as a model to implement such applications.

REFERENCES

[1] Enric Junqué de Fortuny, David Martens, and Foster Provost. Predictive modeling with big data: is bigger really better? *Big Data*, 1(4):215–226, 2013.

[2] Charu C Aggarwal. On k-anonymity and the curse of dimensionality. In *VLDB*, volume 5, pages 901–909, 2005.

[3] Chih-Yi Chiu, Shih-Pin Chao, Ming-Yang Wu, Shi-Nine Yang, and Hsin-Chih Lin. Content-based retrieval for human motion data. *Journal of visual communication and image representation*, 15(3):446–466, 2004.

[4] Vincent W Zheng, Yu Zheng, Xing Xie, and Qiang Yang. Towards mobile intelligence: Learning from gps history data for collaborative recommendation. *Artificial Intelligence*, 184:17–37, 2012.

[5] Hao Ma, Irwin King, and Michael R Lyu. Effective missing data prediction for collaborative filtering. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 39–46, 2007.

[6] Yan V Fyodorov and Boris A Khoruzhenko. Systematic analytical approach to correlation functions of resonances in quantum chaotic scattering. *Physical review letters*, 83(1):65, 1999.

[7] Korbinian Breinl, Giuliano Di Baldassarre, Marc Girons Lopez, Michael Hagenlocher, Giulia Vico, and Anna Rutgersson. Can weather generation capture precipitation patterns across different climates, spatial scales and under data scarcity? *Scientific reports*, 7(1):1–12, 2017.

[8] Harold Hotelling. Relations between two sets of variates. In *Breakthroughs in statistics*, pages 162–190. Springer, 1992.

[9] Thomas H Davenport, Paul Barth, and Randy Bean. How 'big data' is different. 2012.

[10] KU Jaseena, Julie M David, et al. Issues, challenges, and solutions: big data mining. *CS & IT-CSCP*, 4(13):131–140, 2014.

[11] Ioannis Tsamardinos, Giorgos Borboudakis, Pavlos Katsogridakis, Polyvios Pratikakis, and Vassilis Christophides. A greedy feature selection algorithm for big data of high dimensionality. *Machine learning*, 108(2):149–202, 2019.

[12] Lidong Wang. Heterogeneous data and big data analytics. *Automatic Control and Information Sciences*, 3(1):8–15, 2017.

[13] Seref Sagiroglu and Duygu Sinanc. Big data: A review. In *2013 international conference on collaboration technologies and systems (CTS)*, pages 42–47. IEEE, 2013.

[14] Kadir Akbudak, Hatem Ltaief, Aleksandr Mikhalev, Ali Charara, Aniello Esposito, and David Keyes. Exploiting data sparsity for large-scale matrix computations. In *European Conference on Parallel Processing*, pages 721–734. Springer, 2018.

[15] Keith A Brown, Sarah Brittman, Nicolò Maccaferri, Deep Jariwala, and Umberto Celano. Machine learning in nanoscience: big data at small scales. *Nano letters*, 20(1):2–10, 2019.

[16] Michele Ianni, Elio Masciari, Giuseppe Massimo Mazzeo, and Carlo Zaniolo. Clustering goes big: Clubs-p, an algorithm for unsupervised clustering around centroids tailored for big data applications. In *2018 26th Euromicro international conference on parallel, distributed and network-based processing (PDP)*, pages 558–561. IEEE, 2018.

[17] Cristian S Calude and Giuseppe Longo. The deluge of spurious correlations in big data. *Foundations of science*, 22(3):595–612, 2017.

[18] Keith H Coble, Ashok K Mishra, Shannon Ferrell, and Terry Griffin. Big data in agriculture: A challenge for the future. *Applied Economic Perspectives and Policy*, 40(1):79–96, 2018.

[19] Jianqing Fan, Fang Han, and Han Liu. Challenges of big data analysis. *National science review*, 1(2):293–314, 2014.

[20] Valerii Vtoryi, Sergei Vtoryi, Evgenia Lantsova, Vladislav Gordeev, et al. Effect of weather conditions on content of carbon dioxide in barns. In *Proceedings 15th International Scientific Conference «Engineering for rural development*, pages 437–441, 2016.

[21] Sivakiruthika Natchimuthu, Balathandayuthabani Panneer Selvam, and David Bastviken. Influence of weather variables on methane and carbon dioxide flux from a shallow pond. *Biogeochemistry*, 119(1):403–413, 2014.

[22] Johannes Lehmann. A handful of carbon. *Nature*, 447(7141):143–144, 2007.

[23] Onyebuchi A Arah. Bias analysis for uncontrolled confounding in the health sciences. *Annual review of public health*, 38:23–38, 2017.

[24] Katie Seely-Gant and Lisa M Frehill. Exploring bias and error in big data research. *Journal of the Washington Academy of Sciences*, 101(3):29–38, 2015.

[25] Liran Einav and Jonathan Levin. Economics in the age of big data. *Science*, 346(6210), 2014.

[26] Joel Gelernter and Renato Polimanti. Genetics of substance use disorders in the era of big data. *Nature Reviews Genetics*, pages 1–18, 2021.

[27] George Willis and Emmanouil Tranos. Using 'big data'to understand the impacts of uber on taxis in new york city. *Travel Behaviour and Society*, 22:94–107, 2021.

[28] Bernhard Schölkopf, Dominik Janzing, Jonas Peters, Eleni Sgouritsa, Kun Zhang, and Joris Mooij. Semi-supervised learning in causal and anticausal settings. In *Empirical Inference*, pages 129–141. Springer, 2013.

[29] Jiangui Liu, John R Miller, Driss Haboudane, and Elizabeth Pattey. Exploring the relationship between red edge parameters and crop variables for precision agriculture. In *IGARSS 2004. 2004 IEEE International Geoscience and Remote Sensing Symposium*, volume 2, pages 1276–1279. IEEE, 2004.

[30] Driss Haboudane, John R Miller, Nicolas Tremblay, Pablo J Zarco-Tejada, and Louise Dextraze. Integrated narrow-band vegetation indices for prediction of crop chlorophyll content for application to precision agriculture. *Remote sensing of environment*, 81(2-3):416–426, 2002.

[31] Andrew Hall, David W Lamb, Bruno P Holzapfel, and John P Louis. Within-season temporal variation in correlations between vineyard canopy and winegrape composition and yield. *Precision Agriculture*, 12(1):103–117, 2011.

[32] Oliver Müller, Iris Junglas, Jan vom Brocke, and Stefan Debortoli. Utilizing big data analytics for information systems research: challenges, promises and guidelines. *European Journal of Information Systems*, 25(4):289–302, 2016.

[33] David Freedman, Robert Pisani, and Roger Purves. Statistics (international student edition). *Pisani, R. Purves, 4th edn. WW Norton & Company, New York*, 2007.

[34] Bruce Thompson. *Canonical correlation analysis: Uses and interpretation*. Number 47. Sage, 1984.

[35] Thomas R Knapp. Canonical correlation analysis: A general parametric significance-testing system. *Psychological Bulletin*, 85(2):410, 1978.

[36] Robin K Henson. Demystifying parametric analyses: Illustrating canonical correlation analysis as the multivariate general linear model. *Multiple Linear Regression Viewpoints*, 26(1):11–19, 2000.

[37] RJ McKay. Variable selection in multivariate regression: An application of simultaneous test procedures. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(3):371–380, 1977.

[38] Heysem Kaya, Florian Eyben, Albert Ali Salah, and Björn Schuller. Cca based feature selection with application to continuous depression recognition from acoustic speech features. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3729–3733. IEEE, 2014.

[39] Liang Sun, Shuiwang Ji, and Jieping Ye. Canonical correlation analysis for multilabel classification: A least-squares formulation, extensions, and analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):194–200, 2010.

[40] Viresh Ranjan, Nikhil Rasiwasia, and CV Jawahar. Multi-label cross-modal retrieval. In *Proceedings of the IEEE international conference on computer vision*, pages 4094–4102, 2015.

[41] HG Gauch and TR Wentworth. Canonical correlation analysis as an ordination technique. *Vegetatio*, 33(1):17–22, 1976.

[42] Su-Yun Huang, Mei-Hsien Lee, and Chuhsing Kate Hsiao. Kernel canonical correlation analysis and its applications to nonlinear measures of association and test of independence. *Institute of Statistical Science: Academia Sinica, Taiwan*, 2006.

[43] David R Hardoon, Janaina Mourao-Miranda, Michael Brammer, and John Shawe-Taylor. Unsupervised analysis of fmri data using kernel canonical correlation. *NeuroImage*, 37(4):1250–1259, 2007.

[44] Henry C Thode. *Testing for normality*. CRC press, 2002.

[45] Arighna Roy and Anne Denton. Nearest-neighbor-intersection algorithm for identifying strong predictors using high-dimensional data. In *2019 IEEE International Conference on Electro Information Technology (EIT)*, pages 416–421. IEEE, 2019.

[46] Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. Deep canonical correlation analysis. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1247–1255, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.

[47] Viacheslav I Adamchuk, J W Hummel, MT Morgan, and SK Upadhyaya. On-the-go soil sensors for precision agriculture. *Computers and electronics in agriculture*, 44(1):71–91, 2004.

[48] Chenghai Yang, James H Everitt, Qian Du, Bin Luo, and Jocelyn Chanussot. Using high-resolution airborne and satellite imagery to assess crop growth and yield variability for precision agriculture. *Proceedings of the IEEE*, 101(3):582–592, 2012.

[49] Bruno Basso, JT Ritchie, FJ Pierce, RP Braga, and JW Jones. Spatial validation of crop models for precision agriculture. *Agricultural Systems*, 68(2):97–112, 2001.

[50] Khalid A Al-Gaadi, Abdalhaleem A Hassaballa, ElKamil Tola, Ahmed G Kayad, Rangaswamy Madugundu, Bander Alblewi, and Fahad Assiri. Prediction of potato crop yield using precision agriculture techniques. *PloS one*, 11(9):e0162219, 2016.

[51] John Fox. *Nonparametric simple regression: Smoothing scatterplots*. Number 130. Sage, 2000.

[52] Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. Deep canonical correlation analysis. In *International conference on machine learning*, pages 1247–1255, 2013.

[53] Koby Todros and Alfred O Hero. Measure transformed canonical correlation analysis with application to financial data. In *2012 IEEE 7th Sensor Array and Multichannel Signal Processing Workshop (SAM)*, pages 361–364. IEEE, 2012.

[54] Yanping Chen, Eamonn Keogh, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, and Gustavo Batista. The ucr time series classification archive. 2015.

[55] A Denton, E Momsen, J Xu, D Franzen, J Nowatzki, and K Farahmand. Use of quality and quantity information towards evaluating the importance of independent variables in yield prediction. In *Proceedings of the International Conference for Precision Agriculture*, 2014.

[56] Anne M Denton and Arighna Roy. Cluster-overlap algorithm for assessing preprocessing choices in environmental sustainability. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 4212–4220. IEEE, 2017.

[57] Thomas Dunne, Weihua Zhang, and Brian F Aubry. Effects of rainfall, vegetation, and microtopography on infiltration and runoff. *Water Resources Research*, 27(9):2271–2285, 1991.

[58] Koushal Kumar and Gour Sundar Mitra Thakur. Advanced applications of neural networks and artificial intelligence: A review. *IJ Information Technology and Computer Science*, 6:57–68, 2012.

[59] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Marc'aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, Quoc V. Le, and Andrew Y. Ng. Large scale distributed deep networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1223–1231. Curran Associates, Inc., 2012.

[60] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

[61] Tanzeem Choudhury, Gaetano Borriello, Sunny Consolvo, Dirk Haehnel, Beverly Harrison, Bruce Hemingway, Jeffrey Hightower, Karl Koscher, Anthony LaMarca, James A Landay, et al. The mobile sensing platform: An embedded activity recognition system. *IEEE Pervasive Computing*, 7(2):32–41, 2008.

[62] Ajita Rattani and Reza Derakhshani. On fine-tuning convolutional neural networks for smartphone based ocular recognition. In *2017 IEEE International Joint Conference on Biometrics (IJCB)*, pages 762–767. IEEE, 2017.

[63] Vijaykrishnan Narayanan and Yuan Xie. Reliability concerns in embedded system designs. *Computer*, 39(1):118–120, 2006.

[64] Edward A Lee. What's ahead for embedded software? *Computer*, 33(9):18–26, 2000.

[65] Edward A Lee. Cyber physical systems: Design challenges. In *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*, pages 363–369. IEEE, 2008.

[66] Avishek Biswas and Anantha P Chandrakasan. Conv-ram: An energy-efficient sram with embedded convolution computation for low-power cnn-based machine learning applications. In *2018 IEEE International Solid-State Circuits Conference-(ISSCC)*, pages 488–490. IEEE, 2018.

[67] Robert C Baumann. Soft errors in commercial integrated circuits. *International Journal of High Speed Electronics and Systems*, 14(02):299–309, 2004.

[68] Robert C Baumann. Radiation-induced soft errors in advanced semiconductor technologies. *IEEE Transactions on Device and materials reliability*, 5(3):305–316, 2005.

[69] Lin Li, Vijay Degalahal, Narayanan Vijaykrishnan, Mahmut Kandemir, and Mary Jane Irwin. Soft error and energy consumption interactions: A data cache perspective. In *Proceedings of the 2004 international symposium on Low power electronics and design*, pages 132–137. ACM, 2004.

[70] Frank Mueller. Challenges for cyber-physical systems: Security, timing analysis and soft error protection. In *High-Confidence Software Platforms for Cyber-Physical Systems (HCSP-CPS) Workshop, Alexandria, Virginia*, page 4, 2006.

[71] Jose Maiz, Scott Hareland, Kevin Zhang, and Patrick Armstrong. Characterization of multi-bit soft error events in advanced srams. In *IEEE International Electron Devices Meeting 2003*, pages 21–4. IEEE, 2003.

[72] Jaeseok Kim and Hyunchul Shin. *Algorithm & SoC Design for Automotive Vision Systems*. Springer, 2014.

[73] Gopalakrishnan Srinivasan, Parami Wijesinghe, Syed Shakib Sarwar, Akhilesh Jaiswal, and Kaushik Roy. Significance driven hybrid 8t-6t sram for energy-efficient synaptic storage in artificial neural networks. In *2016 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 151–156. IEEE, 2016.

[74] Dipan Kumar Mandal, Jagadeesh Sankaran, Akshay Gupta, Kyle Castille, Shraddha Gondkar, Sanmati Kamath, Pooja Sundar, and Alan Phipps. An embedded vision engine (eve) for automotive vision processing. In *2014 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 49–52. IEEE, 2014.

[75] Ignacio Laguna, Martin Schulz, David F Richards, Jon Calhoun, and Luke Olson. Ipas: Intelligent protection against silent output corruption in scientific applications. In *2016 IEEE/ACM International Symposium on Code Generation and Optimization (CGO)*, pages 227–238. IEEE, 2016.

[76] Guochun Shi, Jeremy Enos, Michael Showerman, and Volodymyr Kindratenko. On testing gpu memory for hard and soft errors. In *Proc. Symposium on Application Accelerators in High-Performance Computing*, volume 107, 2009.

[77] Guanpeng Li, Siva Kumar Sastry Hari, Michael Sullivan, Timothy Tsai, Karthik Pattabiraman, Joel Emer, and Stephen W Keckler. Understanding error propagation in deep learning neural network (dnn) accelerators and applications. In *Proceedings of the International Con-*

ference for High Performance Computing, Networking, Storage and Analysis, page 8. ACM, 2017.

[78] Rizwan A Ashraf, Roberto Gioiosa, Gokcen Kestor, Ronald F DeMara, Chen-Yong Cher, and Pradip Bose. Understanding the propagation of transient errors in hpc applications. In *SC'15: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–12. IEEE, 2015.

[79] Nahmsuk Oh, Philip P Shirvani, and Edward J McCluskey. Error detection by duplicated instructions in super-scalar processors. *IEEE Transactions on Reliability*, 51(1):63–75, 2002.

[80] Augusto Vega, Chung-Ching Lin, Karthik Swaminathan, Alper Buyuktosunoglu, Sharathchandra Pankanti, and Pradip Bose. Resilient, uav-embedded real-time computing. In *2015 33rd IEEE International Conference on Computer Design (ICCD)*, pages 736–739. IEEE, 2015.

[81] Kuang-Hua Huang et al. Algorithm-based fault tolerance for matrix operations. *IEEE transactions on computers*, 100(6):518–528, 1984.

[82] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. In *Acm sigmod record*, volume 22, pages 207–216. ACM, 1993.

[83] Gregory Piatetsky-Shapiro. Discovery, analysis, and presentation of strong rules. *Knowledge discovery in databases*, pages 229–238, 1991.

[84] Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. In *ACM sigmod record*, volume 29, pages 1–12. ACM, 2000.

[85] Hieu Minh Bui, Margaret Lech, Eva Cheng, Katrina Neville, and Ian S Burnett. Object recognition using deep convolutional features transformed by a recursive network structure. *IEEE Access*, 4:10059–10066, 2016.

[86] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do cifar-10 classifiers generalize to cifar-10? *arXiv preprint arXiv:1806.00451*, 2018.

[87] Christian Borgelt. An implementation of the fp-growth algorithm. In *Proceedings of the 1st international workshop on open source data mining: frequent pattern mining implementations*, pages 1–5. ACM, 2005.

[88] Francesco Ricci, Lior Rokach, and Bracha Shapira. Introduction to recommender systems handbook. In *Recommender systems handbook*, pages 1–35. Springer, 2011.

[89] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge & Data Engineering*, (6):734–749, 2005.

[90] Robin Burke. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12(4):331–370, 2002.

[91] Xiaoyuan Su and Taghi M Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in artificial intelligence*, 2009, 2009.

[92] Blerina Lika, Kostas Kolomvatsos, and Stathes Hadjiefthymiades. Facing the cold start problem in recommender systems. *Expert Systems with Applications*, 41(4):2065–2073, 2014.

[93] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295, 2001.

[94] Daniel Billsus and Michael J Pazzani. Learning collaborative information filters. In *Icml*, volume 98, pages 46–54, 1998.

[95] Guibing Guo, Jie Zhang, and Daniel Thalmann. Merging trust in collaborative filtering to alleviate data sparsity and cold start. *Knowledge-Based Systems*, 57:57–68, 2014.

[96] Charu C Aggarwal. An introduction to recommender systems. In *Recommender systems*, pages 1–28. Springer, 2016.

[97] Dietmar Jannach, Lukas Lerche, Iman Kamehkhosh, and Michael Jugovac. What recommenders recommend: an analysis of recommendation biases and possible countermeasures. *User Modeling and User-Adapted Interaction*, 25(5):427–491, 2015.

[98] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[99] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Application of dimensionality reduction in recommender system-a case study. Technical report, Minnesota Univ Minneapolis Dept of Computer Science, 2000.

[100] Carlos A Gomez-Uribe and Neil Hunt. The netflix recommender system: Algorithms, business value, and innovation. *ACM Transactions on Management Information Systems (TMIS)*, 6(4):1–19, 2015.

[101] Dheeraj kumar Bokde, Sheetal Girase, and Debajyoti Mukhopadhyay. Role of matrix factorization model in collaborative filtering algorithm: A survey. *CoRR, abs/1503.07475*, 2015.

[102] Anunay Gupta, Om Prakash Yadav, Arighna Roy, Douglas DeVoto, and Joshua Major. Degradation modeling and reliability assessment of capacitors. In *International Electronic Packaging Technical Conference and Exhibition*, volume 59322, page V001T06A018. American Society of Mechanical Engineers, 2019.

[103] Ameneh Forouzandeh Shahraki, Arighna Roy, Om Prakash Yadav, and Ajay Pal Singh Rathore. Predicting remaining useful life based on instance-based learning. In *2019 Annual Reliability and Maintainability Symposium (RAMS)*, pages 1–6. IEEE, 2019.

[104] Abhishek Srivastava, Pradip Kumar Bala, and Bipul Kumar. New perspectives on gray sheep behavior in e-commerce recommendations. *Journal of Retailing and Consumer Services*, 53, 2020.

[105] Anna Zatevakhina, Natalia Dedyukhina, and Oleg Klioutchnikov. Recommender systems-the foundation of an intelligent financial platform: Prospects of development. In *2019 International Conference on Artificial Intelligence: Applications and Innovations (IC-AIAI)*, pages 104–1046. IEEE.

[106] Quanming Yao, Xiangning Chen, James T Kwok, Yong Li, and Cho-Jui Hsieh. Efficient neural interaction function search for collaborative filtering. In *Proceedings of The Web Conference 2020*, pages 1660–1670, 2020.

[107] Sang-Min Choi and Yo-Sub Han. A content recommendation system based on category correlations. In *2010 Fifth International Multi-conference on Computing in the Global Information Technology*, pages 66–70. IEEE, 2010.

[108] Yashar Deldjoo, Mehdi Elahi, and Paolo Cremonesi. Using visual features and latent factors for movie recommendation. CEUR-WS, 2016.

[109] Yashar Deldjoo, Mehdi Elahi, Massimo Quadrana, and Paolo Cremonesi. Toward building a content-based video recommendation system based on low-level features. In *International Conference on Electronic Commerce and Web Technologies*, pages 45–56. Springer, 2015.

[110] Michael Fleischman and Eduard Hovy. Recommendations without user preferences: a natural language processing approach. In *IUI*, volume 3, pages 242–244. Citeseer, 2003.

[111] Mark Van Setten. Experiments with a recommendation technique that learns category interests. In *ICWI*, pages 722–725, 2002.

[112] Chumki Basu, Haym Hirsh, William Cohen, et al. Recommendation as classification: Using social and content-based information in recommendation. In *Aaai/iaai*, pages 714–720, 1998.

[113] Sakshi Bansal, Chetna Gupta, and Anuja Arora. User tweets based genre prediction and movie recommendation using lsi and svd. In *2016 Ninth International Conference on Contemporary Computing (IC3)*, pages 1–6. IEEE, 2016.

[114] Sang-Ki Ko, Sang-Min Choi, Hae-Sung Eom, Jeong-Won Cha, Hyunchul Cho, Laehyum Kim, and Yo-Sub Han. A smart movie recommendation system. In *Symposium on Human Interface*, pages 558–566. Springer, 2011.

[115] Saúl Vargas, Linas Baltrunas, Alexandros Karatzoglou, and Pablo Castells. Coverage, redundancy and size-awareness in genre diversity for recommender systems. In *Proceedings of the 8th ACM Conference on Recommender systems*, pages 209–216. ACM, 2014.

[116] Shaghayegh Sahebi and William W Cohen. Community-based recommendations: a solution to the cold start problem. In *Workshop on recommender systems and the social web, RSWEB*, page 60, 2011.

[117] R Kiran, Pradeep Kumar, and Bharat Bhasker. Dnnrec: A novel deep learning based hybrid recommender system. *Expert Systems with Applications*, 144:113054, 2020.

[118] P. M. Henriques and J. Mendes-Moreira. Combining recommendation systems with a dynamic weighted technique. In *2016 Eleventh International Conference on Digital Information Management (ICDIM)*, pages 203–208, Sep. 2016.

[119] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19, 2015.