Incorporating Dimension Reduction and Clustering into a Data Processing Pipeline for Improving the Performance of Machine Learning Models

A dimenziócsökkentés és a klaszterezés beépítése az adatfeldolgozási folyamatba a gépi tanulási modellek teljesítményének javításához

Ph.D. Thesis Booklet

Khalid Kahloot

Advisor:
Peéter Ekler, Ph.D.
Kahloot Khalid

Budapest University of Technology and Economics
Faculty of Electrical Engineering and Informatics
Department of Automation and Applied Informatics

1111 Budapest, Műegyetem rkp. 3.

E-mail: khalid.kahloot@aut.bme.hu
Tel: +36(1)4631668
Fax: +36(1)4633478

Advisor:
Péter Ekler, Ph.D.
1 Introduction

The brain of humans cannot comprehend more than one piece of information at a time. Humans tend to condense all that information into one or two dimensions and ignore all the others to process the perceived information as fast as possible. Humans also tend to sort out information and to continuously categorize the perceived information. No wonder that we hear the expression “Where do we draw the line” in the English language while debating a vague topic. Similarly, the visual perception of information in two dimensions is always combined with the tendency to separate the space with lines.

In data analytics, many techniques have been developed to linearly separate the data. Most techniques work by combining dimension reduction and clustering of similar data points, and then projecting the data points onto a representation space where similar data points are linearly separable from other dissimilar data points. The higher the linear separability achieved in the representation space is, the more successful is the technique in terms of visualization, classification, or clustering. Consider the upper row in Figure 1, which presents four randomly-generated datasets in two dimensions. The attempt to fit linear regression lines to the data yields a poor performance, at most, in the classification by colored classes. On the other hand, the lower row of Figure 1 illustrates the datasets in the upper row after applying some sort of transformation to the manifold. The projected manifold is clearly linearly separated by the linear regression lines. These forms of data representation in the projected manifold are the simplest ones and can be learned by any linear classifier with high accuracy.

![Figure 1: Randomly-generated two-dimensional datasets and corresponding linear transformation.](image)

Domain experts curate the datasets that appear in publications and split them into training, testing, and validation sub-datasets. Consequently, machine learning models typically perform well on such split-by-hand datasets; however, splitting real-world datasets into curated sub-datasets (i.e., training, testing, and validation sub-datasets) requires extensive effort. Repetitive random splitting is usually carried out, evaluated, and improved until a better score is reached, according to the evaluation metrics.

In this dissertation, I use two instrumental pipelines to test out traditional and proposed methods. The first instrumental pipeline is a traditional data processing pipeline, which can be divided into data understanding, data preparation, and modeling.
The second instrumental pipeline is the Algorithmic Splitting pipeline, which can be divided into manifold projection, spatial clustering, and data splitting. The theoretical framework of this dissertation pivots around these three. The first is about the role of data analytics in designing the machine learning models. The second is about incorporating dimension reduction and spatial clustering together in a data processing pipeline. The third is about splitting the data based on data statistics to improve the performance of a machine learning model.

2 Summary of the Methodology

In this section, the datasets used are described to give the reader some perspective on the complexity of the challenges that these datasets comprise. In addition, the methods and techniques used in each thesis are described to further inform the reader about these topics.

2.1 Description of the Used Datasets

In this section, I describe the datasets that have been used, which comprise the respiratory, road accidents, traffic, weather, and the famous image datasets. The respiratory dataset was collected by a medical team from the pulmonary department of Szent János Kórházm Hospital in November 2016. The traffic dataset was collected by the California department of transportation. The famous image datasets include a total of 70,000 images with 10 categories and are handwritten digits and fashion objects. The complex datasets are the CIFAR 10 and CIFAR 100, which are sets of 10 and 100 miscellaneous labeled classes, respectively. The SmallNORB dataset, which encloses 48,600 images of fifty toys, was also used.

2.1.1 Respiratory Dataset

In total, 2,254 volunteers were recruited and asked to sit in front of the respiratory device for about 18 minutes for the measurements. This experiment was carried out in a low-noise intensity environment to capture the breaths with a significantly lower noise intensity. The medical team measured three parameters: time, velocity, and volume for each patient. Finally, the dataset was prepared in a triple-structure matrix and it was stored (170 MB). For instance, volunteer identification number 879 has 4,360 trios as structures; more specifically, the sample number 4,000 was captured at 15.9411 s. Furthermore, the velocity was recorded as -1.2693 m/s at this particular time, and the volume was 3.6613 L/s.

2.1.2 Road Accidents and Safety Statistics Dataset

This Dataset\textsuperscript{1} contains the traffic flow and the accidents occurring from 2012 to 2014 in the city of London, UK. It has been compiled by the UK government. Accident events are aggregated into a square grid and stacked vertically. The color of each event is determined

\textsuperscript{1}gov.uk/government/collections/road-accidents-and-safety-statistics
by the number of casualties. The dataset contains 26 features, with the most important being the coordinates, number of vehicles, number of causalities, light, weather conditions, among others. It includes a total of 31,153 records.

2.1.3 Performance Measurements Systems (PeMS) Dataset

The daily traffic flow data until the last month of 2018 were obtained from the California department of transportation website\(^2\). Each document shows the traffic information for the whole day, with 5-minute intervals between each sample. A wide range of parameters is shown in each data file, including the average speed, average travel time, and average free-flow speed. Each sensor is identified by a different source ID.

2.1.4 MesoWest Dataset

MesoWest Project\(^3\) is a public weather website. Several weather stations spread out across the region provide data about different parameters, including the temperature, wind speed, and precipitation. The weather data have also been collected in 5-minute intervals. I selected the active and most up-to-date (at the time of the experiment) weather stations for this work.

2.1.5 Famous Image Datasets

The datasets used were MNIST [LeCun and Cortes, 2010], and Fashion-MNIST [Xiao et al., 2017], both of them consisting of a total of 70,000 28×28-pixel images with ten categories, representing handwritten digits and fashion articles, respectively. The complex datasets CIFAR 100 [Krizhevsky et al., b, 2010], which is a set of miscellaneous 100 labeled classes, and CIFAR-10 [Krizhevsky et al., a, 2010], which is a smaller version with ten classes and 60,000 colored 32×32-pixel images of ten object categories, were used. Moreover, SmallNORB [LeCun et al., 2004], which encloses 48,600 96×96-pixel images of fifty toys, was used. This dataset has many labels; however, the object label with five categories was used. Lastly, 60,000 32×32-pixel images from the Shapes3D [Chris and Hyunjik, 2018] dataset were used, together with their shape label (consisting of four categories). Each dataset was split into two sub-datasets (training and testing sub-datasets), which were either used or concatenated and then randomly split into the training and testing sub-datasets after shuffling. However, if there was no pre-split (no original split) of the dataset, as in the case of Shapes3D, the loaded dataset was only shuffled and split into training and testing sub-datasets.

2.2 Data Understanding Step

I wanted to illustrate a traditional data processing pipeline. A respiratory dataset was subjected to signal processing and data mining techniques. I was able to categorize and characterize nine breathing behaviors in randomly selected volunteers. The goal of this

\(^2\) pemsdot.ca.gov
\(^3\) mesowest.utah.edu
The experiment was to convert the reported data into useful statistics and figures.

Data Sampling was conducted in November 2016 at the Szent János Kórház Hospital, Budapest. The investigation aimed to capture the volume and velocity of the air breathed by the volunteers. Using a respiratory device, the medical team managed to perform 25 readings/s. At each sampling point, the medical team measured the amount of air (in L/s) that flowed into or blew from the mouth of each volunteer. Simultaneously, the medical team measured the speed of the air (in m/s) inhaled or exhaled.

The volunteers were asked to sit in front of a respiratory device for about 18 min for the measurements. The dataset was prepared in a triple-structure matrix, with the time, volume, and velocity values corresponding to each patient using MatLAB software (MathWorks, Inc.).

### 2.2.1 Feature Extraction and Mining

For feature extraction, we exploited a mining algorithm. A modified version of the Apriori algorithm, in which frequency probabilities are represented using a graph data structure, was used for association mining. All rules are composed of nodes that are linked together by in-degree and off-degree edges. To create node weighted and edge weighted directed graphs, the algorithm employs Apriori statistical rule mining.

The Apriori algorithm considers the data as items in a collection of baskets and statistically generates rules for the consequent frequencies of items. A modified version of Apriori considers the dataset as a set of features with a variant set of values. Then, it calculates the weight values that express the probabilistic relationships between them. Like the decision tree rules, the set of associations \( a_1, a_2, \ldots, a_d \) is mapped to a set of graphs \( G_1, G_2, \ldots, G_d \). The modified version of Apriori utilizes the graph data structures immediately in the first step. As presented below, two algorithms should be executed consecutively. First, the algorithm 1 uses parameter \( \text{minSupport} \), which is a threshold of the metric \( \text{Support} \). Second, the algorithm uses the parameter \( \text{minConfidence} \), which is a threshold of the metric \( \text{Confidence} \). The formal definitions of these metrics are:

\[
\text{Support: } s(N_iN_j) = \frac{\sigma(N_j \cup N_i)}{n} \quad (2.1)
\]

\[
\text{Confidence: } c(N_iN_j) = \frac{\sigma(N_j \cup N_i)}{\sigma(N_i)} \quad (2.2)
\]

#### 2.2.2 Graph Generation Using Apriori Association

Let \( F = f_1, f_2, \ldots, f_d \) be the set of all variables and \( V = v_1, v_2, \ldots, v_n \) be the set of all values in a dataset \( D \). The objective of the algorithm 1 is to build a topological structure out of this dataset using association analysis. The topological structure is a weighted multi-dimensional directed graph which contains nodes, which are the features, and directed edges emerging from each node. An important property of an edge is its weight, which refers to the statistical likelihood of occurrence in a certain order. The
support count was used as a weighing scale. Mathematically, the support count, \( \sigma(N) \), for an ordered subset of features \( F \) can be defined as follows:

\[
\sigma(N) = \sum_{i=1}^{n} |\{v_i | f \subseteq v_i, v_i \in V\}|
\]

(2.3)

The objective of the algorithm 1 is to eliminate the weakest edges by using Apriori algorithm to weigh the edges. Moreover, the algorithm aims to reduce the number of comparisons by taking advantage of the graph data structure and algorithms. Let \( N_k \) and \( E_k \) denote the set of nodes and edges, respectively. Initially, the Apriori procedure generates a temporary set of graphs \( G_t \) to represent the set of features, and then determines the weight of each node. Iteratively, the algorithm generates new edges, updates the weight of the existing edges, and uses the procedures combined to generate the support graph \( G_s \).

Algorithm 1 Graph Generation by support.

\[
G_t = G_1, G_2, ..., G_d
\]

\[
N_k = i | i \in I \land \sigma(\{i\}) \geq N_{\text{minSup}}
\]

\[
E_k = \text{Aproiri}(N_k, N_k)
\]

while \( N_k \neq \emptyset \) do

\[
N_{k-1} = \text{subset}(N_k, \text{minSup})
\]

\[
E_{k-1} = \text{Aproiri}(N_{k-1}, N_{k-1})
\]

\[
G_i = \text{addpath}(E_{k-1})
\]

\[
i = i + 1
\]

end while

\[
G_s = \text{combine}(G_t)
\]

2.3 Dealing with Heterogeneous Data

Making a decision based on a single data model can lead to a poor prediction. Stacking a Restricted Boltzmann Machine with a multi-task prediction layer yielded encouraging results: the multi-task neural network's mean square error of the testing dataset significantly improved. Using a backward rolling window to determine the auto-correlation and an optimized window size could yield promising results.

2.3.1 Multi-Task Architectures and Training Approaches

Having deep belief networks (DBNs) as the basic building blocks of my architecture, there were a few different variants that could have been explored. The first one, which was used as a baseline of the comparison with other methods, consists of aggregating a multi-task regression layer on top of the DBN to predict the traffic occupancy of each road. Thus, the upper layer was used to perform a linear regression of the extracted features provided by the output of the DBN.

Using data from different sources and natures is vital to decide how they will be grouped. For instance, the data can be grouped earlier, in the input layer, or in further
layers. With an early fusion, the units can take advantage of the shared weight, but its performance can be diminished if the data do not share the same patterns. The proposed architecture consists of the following steps:

1. Traffic and weather data are trained in an unsupervised manner using two parallel DBNs;
2. Supervised learning can or cannot be applied according to the adopted procedure explained in the next paragraphs;
3. A shared representation of the data is obtained by concatenating the outputs from the previous step;
4. The shared representation is trained in an unsupervised manner using another DBN;
5. Lastly, supervised training is performed through different branches of the architecture, according to the procedures explained in the next paragraphs.

2.3.2 End-to-End Training Approach

This training approach is only performed for unsupervised learning in DBN 1 and DBN 2 before concatenating the outputs. Afterward, unsupervised learning is applied in DBN 3. After including a regression layer on top of DBN 3, optimization is done for all the weight values in the network using Nesterov-accelerated adaptive moment estimation gradient [Dozat, 2016]. The learning rate was set to 0.002, and the batch and momentum gradient coefficients $\beta_1$ and $\beta_2$ were set to 0.9 and 0.999, respectively.

2.3.3 Ensemble of DBN and RNN Approach

The last one is the third approach. A regression layer was added on top of each network to predict the next time step of traffic occupancy and weather conditions. Those two networks were trained in two ways. Partially training those branches ensures that there will be no losses when representing the data previously acquired during learning. The schematics for the third approach is shown in Figure 2.
2.4 Improving the Data Processing via Iterative Spatial Clustering

I searched for ways to improve the resulting visualization of t-SNE by adjusting the hyper-parameters. By adjusting the early exaggeration parameter and employing a heavy-tailed t-distribution kernel, t-SNE generates a linear separable manifold. The number of data points included by spatial clustering was used as a metric for determining how good the projected manifold of each experiment was. This measurement was used to evaluate the t-SNE degree of freedom values.

2.4.1 Manifold Projection Using t-SNE

The most successful version of SNE is the t-SNE [van der Maaten and Hinton, 2008]. Despite the fact that the computational and memory complexity of t-SNE are both $O(n^2)$, it has been used to successfully visualize large real-world datasets with limited computational demands. The datasets in the upper row were shown in the lower row after 50,000 iterations gradient descent optimization using t-SNE. The linear regression lines separate the projected manifold with a straight line. Those types of data in the projected manifold are the simplest and can be learned with high accuracy by any linear classifier.

2.4.2 t-SNE Degrees of Freedom

The FFT kernel transformation worked quite well for MNIST, Fashion-MNIST, and SmallNORB. These datasets have one object per image. The lower row of Figure 10
shows a zoom-in of these. The color of each cluster corresponds to the original color label. For instance, there are three "digit 6" yellow clusters. The t-SNE recognized three patterns in the MNIST of how to write the digit 6. It is very discernible that the resulting clusters are not pure. For instance, the "digit 6" clusters have some "digit 0", "digit 8", and "digit 2" among them. Meanwhile, the use of CIFAR10 and CIFAR100 resulted in a mixture of color-coded data points.

2.4.3 Iterative Spatial Clustering

HDBSCAN algorithm has a feature (i.e., the data points that have an inconstant affinity to other data points are not assigned to any cluster). Thus, to address the Fit-SNE drawbacks mentioned in section 2.4.2, HDBSCAN was used. The proposed techniques to balance visualization and clustering are the following:

- Generate the projected manifold using Fit-SNE with a degree of freedom, $\alpha$, of 0.4;
- Cluster the projected manifold using the HDBSCAN algorithm;
- Remove the unassigned data points from the projected manifold;
- Re-run the Fit-SNE on the reduced dataset;
- Re-generate the clusters using the HDBSCAN algorithm.

2.5 Incorporating Uniform Manifold Projection and Spatial Clustering

A novel algorithmic method for splitting datasets for machine learning models was proposed. For estimating the data distribution, Algorithmic Splitting employs deterministic dimension reduction and density-based clustering techniques. Following that, the dataset is split based on the estimated data distribution. Spatial clustering outperforms distribution-based clustering in the Algorithmic Splitting clustering step. HDBSCAN, in particular, outperforms GMM by 35.65%, on average.

![Figure 3: An illustration of the splitting ranges of a one-dimensional standard Gaussian distribution.](image)
2.5.1 Splitting Objective

Algorithmic Splitting aims to increase the performance of machine learning models. The training sub-dataset comprises not only median resembling data points, but also extreme cases. This type of method would enable to split real-world data, such as healthcare and logistics data, into evenly representative sub-datasets.

The ideal case scenario is when the data are perfectly aligned in a one-dimensional standard Gaussian distribution, which makes the task of splitting into training, validation, and testing sub-datasets easy by sampling them out of each distribution range. The ranges have given names to indicate their vicinity to the mean and the mean plus two standard deviations. The data points in the median range are the median resembling points, which cannot be mistaken or confused by a human spectator (e.g., the eight or the standing tall soldier on the left in Figure 4). Conversely, the data points that belong to the extreme range demand the attention of a human spectator to tell them apart (e.g., the weird-looking eight or the top view of the soldier on the right in Figure 4). In contrast, the data points that belong to the quantile range are neither easily distinguished nor easily confused (e.g., in the middle in Figure 4).

2.5.2 Steps and Parameters of Algorithmic Splitting

A scheme of the Algorithmic Splitting is shown in algorithm 2. The main goal of Algorithmic Splitting is to ensure that the sub-datasets have resembling data distributions to that of the original dataset by avoiding random splitting decisions. Therefore, any machine learning algorithm trained and evaluated on the resulting sub-datasets is thoroughly evaluated. To achieve such a goal, a four-step process must be followed:

- First, a **dimension reduction step** is performed because it is almost impossible to draw conclusions from high-dimensional data. This step must use a deterministic transformation that processes the same results every time for a certain dataset. All
Algorithm 2 Algorithmic Splitting Pseudo-code.

INPUT: Dataset
OUTPUT: Testing, Validation, and Training sub-datasets

Ensure: Testing, Validation, and Training sub-datasets are mutually exclusive
dimension reduction to 2D
labeling by clustering

for cluster in clusters do
    calculate $\mu$ and $\sigma$
    median range upper limit = $\mu + \sigma$
    median range lower limit = $\mu - \sigma$
    extreme range upper limit = $\mu + 2\sigma$
    extreme range lower limit = $\mu - 2\sigma$
    for data point in cluster do
        if between lower limit and upper limit of median range then
            Add data point to median range
        else if between the lower limit and upper limit of the extreme range then
            Add to extreme range
        else
            Add to quantile range
        end if
    end for
end for

Results: Testing, Validation, and Training sub-datasets

following steps are performed on the lower dimension data.

• Second, a clustering step has to be applied to the data points assigned to clusters. The clustering labels are used heuristically throughout the splitting process. Therefore, a dimension reduction step is essential for the algorithmic method, followed by labeling by clustering, before further processing.

• Third, a range demarcation step is taken for each cluster of data points. The mean and standard deviation of each cluster are calculated. The median range has the data points which lay within one standard deviation distance to the mean. The extreme range those which lay within a distance of two standard deviations or more from the mean.

• Four, a forming sub-datasets step takes place. The training, validation, and testing sub- datasets represent 70%, 20%, and 10% of the original dataset . A certain percentage of each range per cluster is sampled without replacement.
3 New Theoretical Results

Thesis I.: Data Analytics Provide Useful Instrumental Insights for Problem Understanding

I have used signal processing and dimension reduction techniques to implement a data understanding step for traditional data processing of a respiratory dataset. The purpose of this experiment was to convert the reported data into useful statistics and figures. Signal processing and data mining techniques were used. I was able to classify and characterize nine different breathing behaviors. I have developed a graph-based version of the Apriori algorithm for association mining, in which the frequency probabilities are represented by a graph data structure. All rules are made up of nodes that are linked together by in-/off-degree edges. The modified algorithm was tested using a variety of traffic datasets. I created a cross-exogenous multi-modeling model for blending the probabilities in order to predict the traffic flow. To predict the probabilities of each target, the model is inputted into a DBN and weighted. Experiments show that combining heterogeneous datasets improves the traffic flow prediction accuracy.

Publications related to this thesis: [2], [3], [4], [7], and [8].

Subthesis I.1.: Dimension Reduction and Clustering Demonstrates Categories of Behavior within the Dataset

I implemented a data understanding step for traditional data processing by employing signal processing and dimension reduction techniques on a respiratory dataset. My motivation for conducting this experiment was to convert reported data into useful statistics and figures. I used signal processing and data mining techniques. I was able to categorize and characterize nine breathing behaviors.

Clustering Similar Data Points
After plotting the singular value decomposition (SVD) results in three dimensions, we realized that the data points were aligned spatially, not spherical, and with too much noise. Accordingly, we applied the density-based spatial clustering of applications (DBSCAN) algorithm. The clusters are shown in two and three dimensions in Figure 5.

Discussion and Analysis
A specific breathing behavior for a sample from cluster 6 is shown in Figure 6(a). The volume has dropped to zero many times, which means that the patient is choking. Figure 6(b) shows the velocity difference corresponding to Figure 6(a). The volume hits a local minimum at -2 very rapidly and then increases very fast to a local maximum of 1.6. As shown in Figure 6 and 7, the clusters are characterized by the shape and local values. For instance, for cluster 1, the volume is constantly increasing within a narrow velocity range, as shown in Figure 7(a). Conversely, for cluster 6, the velocity almost reaches zero at a volume of 3 L/s, and then inhaling of the same volume at the same velocity was repeated for two rounds, as shown in Figure 6(a).
Figure 5: Selected features are plotted in two (out of three) dimensions and each color represents a cluster, as per DBSCAN results.

The medical team has a different point of view when analyzing pre-diagnoses and reorders. I provide an analysis from a data science perspective on how the medical team can utilize these figures. According to the medical experience, they can compare these results with those of previously-diagnosed patients. Figure 7(a) plots the velocity versus volume and highlights the drop in velocity in red and its steady increase with the volume increase. Figure 7(b) shows the velocity difference corresponding to the red line in (a). The volume hits a local minimum at -1; however, no local maximum was registered.

Figure 6: Breathing behavior for a sample from cluster 6.
Comparison with the Literature

In Figure 8, we present a comparison of inhaling behaviors. As shown in Figure 8(a), we plotted the volume of the inhaled air for cluster number 2. The lowest part has a smooth increasing and decreasing cycle. The breathing behavior of COPD patients has been documented in a medical article [Gershon et al., 2010]. As shown in Figure 8(b), the air flow of COPD patients is initially limited during quiet breathing, and as the lung volume increases, a new steady state is achieved at a higher lung volume, at which the exploratory flows are more significant than those at the initial inhaling. Figure 8(a) shows the volume behavior having a steady value from 12 to 22 s. Figure 8(b) shows the raid breathing behavior of COPD patients.

Subthesis I.2.: Graph-Based Association Mining Captures the Direct and Indirect Relationships within the Dataset

I have introduced a graph-based version of the Apriori algorithm for association mining, in which the frequency probabilities are represented using a graph data structure. All rules are composed of nodes, which are interconnected by in- and off-degree edges. A modified algorithm was tested based on heterogeneously composed traffic datasets.
Testing the Memory Efficiency

Apriori algorithm has an enormous number of variations that modified the data structure to outperform the original algorithm. The data structure of GAM is quite different from other variations of the Apriori algorithm. Although GAM does not address the performance issues, comparing its performance with that of well-known variant implementations is essential. A survey and comparison are presented in [Bodon et al., 2006]. All tests were carried out on ten public benchmark databases which were downloaded from [Bart and J, 2003]. First, we compared GAM for storing filtered transactions against a sorted list, a red-black tree (B-tree), and a trie.

Table 1: Memory needed as sorting frequencies for the T40I10D100K dataset.

<table>
<thead>
<tr>
<th>min_freq</th>
<th>GAM</th>
<th>Sorted list</th>
<th>B-tree</th>
<th>trie</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.05</td>
<td>10.1</td>
<td>9.3</td>
<td>10.8</td>
<td>55.4</td>
</tr>
<tr>
<td>0.02</td>
<td>12.7</td>
<td>12.7</td>
<td>14.1</td>
<td>70.3</td>
</tr>
<tr>
<td>0.0073</td>
<td>16.3</td>
<td>19.5</td>
<td>20.3</td>
<td>80.3</td>
</tr>
<tr>
<td>0.006</td>
<td>21.8</td>
<td>21.3</td>
<td>21.5</td>
<td>88.4</td>
</tr>
</tbody>
</table>

As shown in Table 1, regarding memory, complex data structures are in distress. The dataset contains the accidents occurring from 2012 to 2014 in London. The close competitor of GAM is the trie implementation and it even overcomes the GAM, especially at high frequencies. However, the added structures of nodes and edges are critical, and we did not mean to optimize the memory. The number of casualties influenced the color of each event. This map was developed by a professional pythonist called Dave FisherHickey and it was published on the Kaggle website. The available data describe the average annual daily flow, which tracks how much traffic there was on all major roads, in addition to accident data from police reports.

Data Mining Use Case for Finding Indirect Relations

The modified algorithm GAM scanned through the features and values to produce 22 (feature, value) pairs with the corresponding weights. For instance, the feature "light conditions" was chosen twice with "daylight" and "darkness" with 3,20225 and 0,5885 causalities, respectively. The interpretation can be that accidents are 3 times more likely to happen in daylight than in the darkness, but the light condition is still the most significant feature. Another example is the "speed limit" that was chosen to be "30" and that implies that 52% of the accidents happened on roads with a speed limit of 30. Likewise, we can state that accidents occur in "Fine without high winds" weather conditions with a probability of 61.8%.

To study the "Number of Casualties", the highest frequent value is 1. The node of "Number of Casualties = 1" and has 5,937 out edges for neighbors:

- 2nd Road Number = 0,
- Urban/Rural Area = 1,
- 1st Road Class = 6,
- Road Number = 0,
- Human Control = None within 50 m,
The values listed above are direct co-occurring values for an accident with a "Number of Casualties = 1". As a summary of the GAM results, the number of possible scenarios is 7,836, which is also the number of in-edges into this node. The edges are causes of accidents when the "Number of Casualties = 1" and 60.5% of them are causes of accidents when the "2nd Road Number = 0". The pair (Speed limit = 30, Number of Casualties = 1) has 404 edges in between. Edges were positioned in 946 dimensions for the pair (Number of Casualties = 1, Road Type = Single carriageway).

Subthesis I.3.: Fusing Multi-Task Neural Network for Heterogeneous Data sources Improves the Forecasting

I developed a cross-exogenous multi-model for blend probabilities for predicting the traffic flow. The model is inputted to a DBN and is weighted to predict the probabilities of each target. Experiments show that combining heterogeneous datasets results in higher accuracy in traffic flow prediction.

Ensembling the DBN and RNN Approaches
In order to assure that DBN 1 and 2 give out a prediction representation instead of only a data representation, a regression layer was added on top of each network to predict the next time step of traffic occupancy and weather conditions, respectively. The Extended DBNs with Regression are trained in two ways. Firstly, they are fully trained until the validation loss started to worsen compared to the previous epochs. Secondly, they are partially trained with ten epochs. Partially training those branches ensures that there will be no loss in the data representation previously acquired during learning. The schematics of the third approach is shown in Figure 2.

Metrics and Evaluation
In this section, we present the results obtained using the different approaches proposed in this work. The metrics chosen to evaluate the regression were the Mean Square Error (MSE) (Equation 3.1) and Mean Absolute Error (MAE) (Equation 3.2), where i is the index throughout the events in the set and j is the index throughout the roads in question:
\[ MSE = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{X} (Y_{i,j} - \hat{Y}_{i,j})^2 \]  

\[ MAE = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{X} |Y_{i,j} - \hat{Y}_{i,j}| \]  

Table 2 shows the error of the training and validation sets for the proposed architectures. In this case, the training and validation sets were split temporally, with the first four and a half months being dedicated to training and the remaining one and a half months to testing.

<table>
<thead>
<tr>
<th>architecture</th>
<th>MSE Train</th>
<th>MAE Train</th>
<th>MSE Test</th>
<th>MAE Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train all DBN</td>
<td>7.334x10^{-4}</td>
<td>1.508x10^{-1}</td>
<td>6.863x10^{-4}</td>
<td>1.495x10^{-1}</td>
</tr>
<tr>
<td>End-to-End Training approach</td>
<td>9.540x10^{-4}</td>
<td>2.073x10^{-1}</td>
<td>2.416x10^{-4}</td>
<td>3.200x10^{-1}</td>
</tr>
<tr>
<td>Greedy training approach</td>
<td>2.175x10^{-4}</td>
<td>7.342x10^{-3}</td>
<td>2.113x10^{-4}</td>
<td>7.700x10^{-1}</td>
</tr>
<tr>
<td>Ensemble approach</td>
<td>2.03x10^{-4}</td>
<td>6.791x10^{-3}</td>
<td>2.201x10^{-4}</td>
<td>7.224x10^{-3}</td>
</tr>
</tbody>
</table>

**Forecasting Future Traffic**

The architecture predicts the traffic for 58 different roads, and we chose two roads to illustrate its performance. Figure 9 shows the real values versus the predicted values, using the architecture with the best overall efficacy, which is the ensemble of DBNs and RNN. Figure 9(a) shows the prediction for the first two days in the training set in 101 South road and Figure 9(b) displays the forecast for two days in the testing set in 101 North road.

![Figure 9: Forecast and Real Values in the (a) Training Set and (b) Testing Set.](image)
Table 3: The number of unclustered data points obtained via HDBSCAN and their percentage related to the size of the datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>$\alpha = 0.2$</th>
<th>$\alpha = 0.4$</th>
<th>$\alpha = 0.6$</th>
<th>$\alpha = 0.8$</th>
<th>$\alpha = 1.0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>0.832%</td>
<td>0.691%</td>
<td>0.830%</td>
<td>1.232%</td>
<td>1.766%</td>
</tr>
<tr>
<td>Fashion-MNIST</td>
<td>1.835%</td>
<td>1.632%</td>
<td>1.632%</td>
<td>4.01%</td>
<td>4.456%</td>
</tr>
<tr>
<td>SmallNORB</td>
<td>2.358%</td>
<td>2.061%</td>
<td>3.634%</td>
<td>4.95%</td>
<td>5.483%</td>
</tr>
<tr>
<td>CIFAR10</td>
<td>6.253%</td>
<td>5.796%</td>
<td>8.097%</td>
<td>15.24%</td>
<td>16.560%</td>
</tr>
<tr>
<td>CIFAR100</td>
<td>6.863%</td>
<td>4.437%</td>
<td>12.908%</td>
<td>13.45%</td>
<td>14.769%</td>
</tr>
</tbody>
</table>

**Thesis II.: Iterative Manifold Projection and Spatial Clustering Results in Well-Representative and Ecstatic Manifold**

I have shown that t-SNE generates a linear separable manifold when the degree of freedom is adjusted. The number of data points obtained via spatial clustering was used as a metric for evaluating the experiments. HDBSCAN outperforms all the other clustering algorithms in terms of clustering accuracy, purity, and V-measure score. I presented an improved t-SNE version with a hierarchical density-based spatial clustering method for neighborhood embedding. The HDBSCAN version iteratively works until it achieves a good visualization of the manifold. In addition, a novel algorithmic method for splitting datasets for machine learning models has been proposed. Spatial clustering outperforms the distribution-based clustering in the Algorithmic Splitting clustering step. HDBSCAN outperforms GMM in terms of accuracy.

Publications related to this thesis: [1], [5], and [6].

**Subthesis II.1.: Projecting Using a Heavy-tailed Distribution Improves the Visualization of the Projected Manifold**

I have shown that the heavy-tailed t-SNE is capable of projecting a better manifold compared to that projected using a normal-tailed t-SNE.

**Evaluation via Spatial Clustering**

I counted the number of unclustered data points and studied its relationship with the degree of freedom of the t-SNE algorithm. I used the count of unclustered data points as an indicator of the quality of the projection of the manifold via t-SNE. In other words, the fewer the data points left unclustered, the better the quality of the t-SNE neighborhood embedding. Even in the visual plot of the projected manifold, clusters are not that dense and lots of data points are scattered away from clusters when using t-SNE with a high value of the degree of freedom.

**Comparing Results Based on the Degree of Freedom**

A comparison between the number of unclustered data points obtained when using multiple degrees of freedom was performed. The default degree of freedom value is 1. When the value of the degree of freedom is smaller, the t-distribution is a heavy-tailed distribution. As shown in Table 3, the maximum number of clustered points achieved using HDBSCAN is for an $\alpha = 0.4$. The normal-tailed t-distribution with an $\alpha = 1.0$ yielded a 32.4% lower number of clustered data points in the projected manifold than that obtained for an $\alpha = 0.4$. Controversially, when the value of the degree of freedom
goes below 0.4, the number of clustered data does not increase anymore. After repeating the experiments 1,000 times and averaging the results, I concluded that the number of data points that have been included is, on average, 32.4% higher for a degree of freedom of 0.4, compared to a degree of freedom of 1.0.

Subthesis II.2.: Iterative Dimension Reduction and Clustering Improves Both Visualization and Clustering

I have introduced a modified version of t-SNE that is better in both visualization and clustering. I have shown that after each iteration of manifold projection using heavy-tailed t-SNE, followed by clustering using HDBSCAN, the clustering metrics have been improved.

Clustering the Projected Manifold
Seven clustering algorithms were applied on the projected manifold: K-means, Spectral clustering, ward hierarchical clustering, birch clustering, Gaussian mixture model, DBSCAN, and HDBSCAN. The objective was to find the most conventional clusters which aggregate the data points and enable linear separations. Clustering purity is a vital metric, which measures how homogeneously composed the clusters are. Clustering accuracy is another metric which rationalizes the generated labels by clustering the original labels of the corresponding datasets. The V-measure metric is the harmonic mean between the homogeneity and completeness of the clustering.

The winner algorithm is the one that scores the highest values for each metric. Scores are detailed in Table 4. For the MNIST dataset, the K-means clustering algorithm was the winner. For the Fashion-MNIST, SmallNORB, and CIFAR100 datasets, HDBSCAN was the winner. For the CIFAR10 dataset, the Gaussian mixture algorithm was the winner. The high runner is the HDBSCAN algorithm with outstanding scores, hence its density-based spatial behavior of the constructed clusters.

Iterative Spatial Clustering
To address the Fit-SNE drawbacks mentioned in section 2.4.2, HDBSCAN algorithm was used due to its feature (i.e., the data points that have an inconstant affinity to other data points are not assigned to any cluster). The proposed techniques to balance visualization and clustering are the following:

- Generate the projected manifold via Fit-SNE with a degree of freedom, \( \alpha \) of 0.4;
- Cluster the projected manifold using the HDBSCAN algorithm;
- Drop the unassigned data points out of the projected manifold;
- Re-run the Fit-SNE on the reduced dataset;
- Re-generate the clusters using the HDBSCAN algorithm.

Comparing the Quality of Iterations
Figure 10 and Figure 11 show the results of the above techniques. The removal of the unassigned data points assisted Fit-SNE in the second round to overcome the drawbacks.
Table 4: Clustering metrics for the target datasets for original labeled classes.

<table>
<thead>
<tr>
<th>Method</th>
<th>MNIST</th>
<th>fMNIST</th>
<th>SNORB</th>
<th>CIFAR10</th>
<th>CIFAR100</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>K-means</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of Clusters</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>10</td>
<td>100</td>
</tr>
<tr>
<td>Clustering Accuracy</td>
<td>0.809</td>
<td>0.565</td>
<td>0.447</td>
<td>0.227</td>
<td>0.081</td>
</tr>
<tr>
<td>Clustering Purity</td>
<td>0.856</td>
<td>0.569</td>
<td>0.450</td>
<td>0.237</td>
<td>0.087</td>
</tr>
<tr>
<td>V-Measure Score</td>
<td>0.843</td>
<td>0.572</td>
<td>0.223</td>
<td>0.090</td>
<td>0.149</td>
</tr>
<tr>
<td><strong>Spectral Clustering</strong></td>
<td>MNIST</td>
<td>fMNIST</td>
<td>SNORB</td>
<td>CIFAR10</td>
<td>CIFAR100</td>
</tr>
<tr>
<td>Number of Clusters</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>10</td>
<td>100</td>
</tr>
<tr>
<td>Clustering Accuracy</td>
<td>0.275</td>
<td>0.210</td>
<td>0.460</td>
<td>0.195</td>
<td>0.081</td>
</tr>
<tr>
<td>Clustering Purity</td>
<td>0.297</td>
<td>0.217</td>
<td>0.460</td>
<td>0.200</td>
<td>0.091</td>
</tr>
<tr>
<td>V-Measure Score</td>
<td>0.280</td>
<td>0.138</td>
<td>0.337</td>
<td>0.080</td>
<td>0.153</td>
</tr>
<tr>
<td><strong>Gaussian Mixture</strong></td>
<td>MNIST</td>
<td>fMNIST</td>
<td>SNORB</td>
<td>CIFAR10</td>
<td>CIFAR100</td>
</tr>
<tr>
<td>Number of Clusters</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>10</td>
<td>100</td>
</tr>
<tr>
<td>Clustering Accuracy</td>
<td>0.821</td>
<td>0.594</td>
<td>0.047</td>
<td>0.226</td>
<td>0.081</td>
</tr>
<tr>
<td>Clustering Purity</td>
<td>0.797</td>
<td>0.630</td>
<td>0.887</td>
<td>0.235</td>
<td>0.085</td>
</tr>
<tr>
<td>V-Measure Score</td>
<td>0.799</td>
<td>0.603</td>
<td>0.380</td>
<td>0.089</td>
<td>0.147</td>
</tr>
<tr>
<td><strong>DBSCAN</strong></td>
<td>MNIST</td>
<td>fMNIST</td>
<td>SNORB</td>
<td>CIFAR10</td>
<td>CIFAR100</td>
</tr>
<tr>
<td>Number of Clusters</td>
<td>200</td>
<td>210</td>
<td>656</td>
<td>648</td>
<td>539</td>
</tr>
<tr>
<td>Clustering Accuracy</td>
<td>0.276</td>
<td>0.166</td>
<td>0.641</td>
<td>0.095</td>
<td>0.086</td>
</tr>
<tr>
<td>Clustering Purity</td>
<td>0.276</td>
<td>0.767</td>
<td>0.887</td>
<td>0.331</td>
<td>0.119</td>
</tr>
<tr>
<td>V-Measure Score</td>
<td>0.683</td>
<td>0.515</td>
<td>0.480</td>
<td>0.136</td>
<td>0.205</td>
</tr>
<tr>
<td><strong>HDBSCAN</strong></td>
<td>MNIST</td>
<td>fMNIST</td>
<td>SNORB</td>
<td>CIFAR10</td>
<td>CIFAR100</td>
</tr>
<tr>
<td>Number of Clusters</td>
<td>100</td>
<td>100</td>
<td>500</td>
<td>332</td>
<td>69</td>
</tr>
<tr>
<td>Clustering Accuracy</td>
<td>0.273</td>
<td>0.317</td>
<td>0.741</td>
<td>0.101</td>
<td>0.081</td>
</tr>
<tr>
<td>Clustering Purity</td>
<td>0.966</td>
<td>0.972</td>
<td>0.987</td>
<td>0.323</td>
<td>0.085</td>
</tr>
<tr>
<td>V-Measure Score</td>
<td>0.677</td>
<td>0.723</td>
<td>0.721</td>
<td>0.130</td>
<td>0.156</td>
</tr>
</tbody>
</table>

mentioned. After deliberation and experimentation, the alpha was set to 0.4. This selection makes t-SNE apply a heavy-tailed kernel transformation.
HDBSCAN has outperformed all of the other clustering algorithms in clustering accuracy, clustering purity, and V-measure score. The Gaussian mixture algorithm was competitive with HDBSCAN and DBSCAN, but only for some datasets, such as the MNIST and Fashion-MNIST. Another experiment has been performed to evaluate the iterative spatial clustering to improve t-SNE visualization. By using t-SNE and HDBSCAN, the projected manifold had well-defined clusters. The number of clustered points obtained using HDBSCAN increases with each iteration, as shown in Table 3.
Table 5: The number of data points that have been unclustered using HDBSCAN for each iteration.

<table>
<thead>
<tr>
<th></th>
<th>iteration 1</th>
<th>iteration 2</th>
<th>iteration 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>0.691%</td>
<td>0.110%</td>
<td>0.2%</td>
</tr>
<tr>
<td>Fashion-MNIST</td>
<td>1.632%</td>
<td>1.380%</td>
<td>1.230%</td>
</tr>
<tr>
<td>SmallNORB</td>
<td>2.061%</td>
<td>1.681%</td>
<td>1.404%</td>
</tr>
<tr>
<td>CIFAR10</td>
<td>5.796%</td>
<td>4.791%</td>
<td>3.207%</td>
</tr>
<tr>
<td>CIFAR100</td>
<td>4.437%</td>
<td>3.744%</td>
<td>2.365%</td>
</tr>
</tbody>
</table>

Table 6: Accuracy results of Algorithmic Splitting using HDBSCAN vs. GMM.

<table>
<thead>
<tr>
<th></th>
<th>Ground Truth</th>
<th>Random Splitting</th>
<th>HDBSCAN Splitting</th>
<th>GMM Splitting</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>79.0%</td>
<td>58.4%</td>
<td>80.6%</td>
<td>79.9%</td>
</tr>
<tr>
<td>Fashion-MNIST</td>
<td>77.8%</td>
<td>33.2%</td>
<td>81.5%</td>
<td>78.9%</td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>76.8%</td>
<td>29.6%</td>
<td>72.3%</td>
<td>65.2%</td>
</tr>
<tr>
<td>SmallNORB</td>
<td>79.4%</td>
<td>27.3%</td>
<td>80.5%</td>
<td>30.8%</td>
</tr>
<tr>
<td>Shapes3D</td>
<td>72.3%</td>
<td>32.6%</td>
<td>76.7%</td>
<td>35.5%</td>
</tr>
</tbody>
</table>

Subthesis II.3.: Incorporating Spatial Clustering with Projection-Based Dimension Reduction Eliminates Noise in Clustering

I have shown that the best performing clustering algorithm on the projected manifold is HDBSCAN because it is spatial and robust to noise.

Comparing HDBSCAN and GMM

Table 6 shows the results of the experimental setup using HDBSCAN and GMM clustering algorithms. The ground truth splits are supposed to be the baseline for all the Algorithmic Splitting setups. Nonetheless, the Algorithmic Splitting method still performed better than the ground truth and random splitting. It is worth mentioning that the accuracy of the models is much worse using the SmallNORB and Shapes3D than when using the MNIST, Fashion-MNIST, or CIFAR-10 datasets. Experiments have shown that HDBSCAN is, on average, 35.65% more accurate than GMM. Therefore, I conclude that HDBSCAN is better than GMM (for clustering) in the clustering step of the Algorithmic Splitting pipeline when incorporated with UMAP for manifold projection.
Thesis III.: Incorporating Uniform Manifold Projection and Spatial Clustering Improves the Performance of Machine Learning Models

I have developed a method for defining statistically distinguishable ranges within a given dataset. I have shown that sampling from statistically distinguishable ranges outperforms ground truth and random sampling. The Algorithmic Splitting pipeline’s training, testing, and validation sub-datasets have been analyzed. I have discovered that the percentages of samples taken from each range vary in terms of classification accuracy. The optimal percentages of sampling from each range are: one-half from the extreme range, one-third from the quantile, and the rest from the median range. I have demonstrated that the Algorithmic Splitting’s training, testing, and validation sub-datasets are robust when used on biased datasets. I have shown that the accuracy of the machine learning models outperforms ground truth and random sampling. I have concluded that the Algorithmic Splitting pipeline’s training, testing, and validation sub-datasets improve the performance of machine learning models.

Publications related to this thesis: [1] and [6].

Subthesis III.1.: Splitting the Dataset by Sampling from statistically Designated Ranges Outperforms Sampling Based on Ground Truth and Random Sampling

I concluded that when forming the training, testing, and validation sub-datasets of the Algorithmic Splitting pipeline, sampling from statistically different ranges outperforms ground truth and random sampling in terms of accuracy.

Forming Sub-Datasets

To overcome any existing bias in the sub-datasets, a percentage-wise random sampling from each range was performed and distributed across the sub-dataset. Sampling was used to define the cut points and divide the dataset’s regions into three main ranges: median, quantile, and extreme ranges (parameters $\alpha$, $\beta$, and $\gamma$, respectively). The use of sampling over other methods allows us to deal with diverse distributions, which is the case in this work, despite the fact that this simplicity relatively compromises the accuracy of the results.

Examples from the median range are typical and easy to recognize, as shown in Figure 12. The examples from the quantile range are somewhat recognizable but less definitive than those from the median range. Examples from the extreme range are confusing and less well-formed than those from other ranges.

The data points from the median range resemble clear handwritten digits, for the MNIST dataset. For the SmallNORB dataset, the figures of the soldier are straight, and his hands can be visualized. The extreme range data points have ill-shaped handwritten digits and the soldier figures are unrecognizable because of their pose.
Table 7: Accuracy results of Algorithmic Splitting using HDBSCAN.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Ground Truth</th>
<th>Random Split</th>
<th>Algorithmic Splitting</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>79.0%</td>
<td>58.4%</td>
<td>80.6%</td>
</tr>
<tr>
<td>Fashion-MNIST</td>
<td>77.8%</td>
<td>33.2%</td>
<td>81.5%</td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>76.8%</td>
<td>29.6%</td>
<td>72.3%</td>
</tr>
<tr>
<td>SmallNORB</td>
<td>79.4%</td>
<td>27.3%</td>
<td>80.5%</td>
</tr>
<tr>
<td>Shapes3D</td>
<td>72.3%</td>
<td>32.6%</td>
<td>76.7%</td>
</tr>
</tbody>
</table>

Figure 12: Examples are shown from each range of the Algorithmic Splitting.

Comparing to Random Splitting and Ground Truth

Table 7 shows the results of the HDBSCAN experimental setup. The ground truth splits outperformed all the Algorithmic Splitting setups. Nonetheless, the Algorithmic Splitting method still performed better than random splitting. The samples from statistically different ranges form part of the training, testing, and validation sub-datasets of the Algorithmic Splitting pipeline. The accuracy of split data via sampling from these ranges outperforms ground truth and random sampling. The experiments have shown that sampling from such ranges can improve computer algorithms by outperforming ground truth and random sampling by 5.4% and 16.3%, respectively.
Subthesis III.2.: Sampling Most of the Sub-Dataset out the Extreme Range Maximizes the Classification Accuracy of Machine Learning Models

I have developed a grid search mechanize for finding out the best combination of the three possible sampling percentages. The classification accuracy has been used as a metric for evaluation. I have concluded that sampling at least half of the sub-dataset out of the extreme range of the data maximizes the classification accuracy of machine learning models.

Comparing HDBSCAN and GMM
After performing all the experiments mentioned above on the five different datasets, interesting findings were obtained and there are plenty of future work possibilities. The best performing hyper-parameters were obtained in the experiments on combinations of the hyper-parameters. The overwhelming winning values of the hyper-parameters are: 35% out of the median range, 10% out of the quantile range, and 55% out of the extreme range. The random splitting experiment results have shown a poor performance compared with the ground truth results.

![Figure 13: Box plot for comparing the winning experiments. Algorithmic Splitting using HDBSCAN shows a higher variance and a more balanced distribution than Algorithmic Splitting using GMM.](image)

The Optimal Percentages
The optimal percentages of sampling out of each range are 10%, 55%, and 35% for the median, quantile, and extreme ranges, respectively. Algorithmic Splitting (with spatial clustering and the above percentages) has shown a uniform distribution of accuracy scores and higher mean values than the distribution-based clustering in all experiments, even in those of bias sampling out the extreme and quantile ranges.
Subthesis III.3.: Algorithmic Splitting is Robust with Biased Datasets and Outperforms Sampling Based on Random Sampling and Ground Truth

I have demonstrated that the Algorithmic Splitting’s training, testing, and validation sub-datasets are robust on biased datasets. I have shown that the accuracy of the machine learning models outperforms random sampling and ground truth.

Comparing HDBSCAN and GMM on Biased Datasets
The box plot in Figure 14 shows a big difference between the mean and the variance of the model prediction of the training, testing, and validation sub-datasets split using Algorithmic Splitting and both HDBSCAN and GMM. The reason for such consistency is that the sub-datasets that these models are trained on are evenly representative of the underlying data distribution.

![Figure 14: Accuracy of the testing sub-dataset for biased sampling percentages and for all datasets.](image)

Machine Learning Performance on Biased Datasets
The accuracy score was still relatively good in all the constructed biased datasets, as shown in Figure 14. HDBSCAN led to a higher accuracy compared with GMM. Although the accuracy variance obtained using HDBSCAN is broader than that obtained using GMM, the accuracy distribution of HDBSCAN is uniform, and the mean accuracy is higher. The average accuracy of the training sub-dataset was 69.6%, and that of the testing sub-dataset was 53.7% for the SmallNORB dataset. In addition to the testing accuracy, Figure 15 shows that Algorithmic Splitting is stable on all the training, validation, and testing sub-datasets.
Table 8: Accuracy results of Algorithmic Splitting on biased datasets.

<table>
<thead>
<tr>
<th></th>
<th>Ground Truth</th>
<th>Random Split</th>
<th>Algorithmic Splitting</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>39.5%</td>
<td>17.5%</td>
<td>70.3%</td>
</tr>
<tr>
<td>Fashion-MNIST</td>
<td>36.4%</td>
<td>9.96%</td>
<td>71.3%</td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>35.2%</td>
<td>8.8%</td>
<td>61.2%</td>
</tr>
<tr>
<td>SmallNORB</td>
<td>32.7%</td>
<td>8.1%</td>
<td>71.4%</td>
</tr>
<tr>
<td>Shapes3D</td>
<td>37.5%</td>
<td>10.7%</td>
<td>67.3%</td>
</tr>
</tbody>
</table>

Figure 15: Accuracy of the SmallNORB training, testing, and validation sub-datasets for the bias sampling percentages.

Comparing to Random Splitting and Ground Truth

Table 8 shows the results of the experimental setup for models that have been trained, validated, and tested on sub-datasets prepared using Algorithmic Splitting and HDBSCAN. The objective of this section is to compare the model performance when using full-size datasets with that using biased datasets. The baseline for this comparison has been set in Table 7. The ground truth splits have dropped to one-half of the accuracy they used to have compared to the baseline. Same-wise, the accuracy of the random splits has dropped to one-third, compared to the baseline. On the other hand, the performance of the Algorithmic Splitting splits has dropped to only 10%. Experiments have shown that the accuracy of the split data via sampling via Algorithmic Splitting outperformed the accuracy of the split data via sampling based on ground truth by 40.6%. In addition, the accuracy of the split data obtained via sampling via Algorithmic Splitting outperformed the accuracy of the split data obtained via random sampling by 75.3%. I conclude that by forming the training, testing, and validation sub-datasets of the biased datasets, the accuracy of the machine learning models outperforms ground truth and random sampling.
4 Summary and Applications

In the first thesis, I propose to combine and incorporate off-the-box dimension reduction and clustering techniques as part of the traditional data processing pipeline. In the data understanding step of the traditional data process, the categories of behavior can be explained by studying only the cluster centroids produced by using DBSCAN after applying SVD. In the feature extraction and mining step of the traditional data processing pipeline, graph-based Apriori association mining captures the direct and the indirect relationships within the data. For dealing with heterogeneous data sources, I have built a multi-task deep neural network on top of DBNs and gated recurrent unit networks, the prediction of co-occurring events is improved.

In the second thesis, I have investigated methods for incorporating dimension reduction and clustering to improve the data processing pipeline. For that, I have tweaked the hyper-parameters of the used algorithms to enable the incorporation of clustering methods into the neighborhood embedding process. Both techniques play an essential role in the data prepossessing of any machine learning algorithm.

In the third thesis, I propose a novel Algorithmic Splitting method for preparing datasets for machine learning models. In the dimension reduction step of the Algorithmic Splitting pipeline, manifold generation using UMAP always results in a deterministic manifold for the same dataset. In the clustering step of the Algorithmic Splitting pipeline, by labeling the data points using HDBSCAN, each data point is assigned to statistically different ranges. In the range demarcation for the projected manifold step of the Algorithmic Splitting pipeline, a manifold approximation using projection-based dimension reduction can be well integrated with spatial density-based clustering. The clusters are well-formed and linearly separable.

5 Scientific Publications

International Journals


International Conference Proceedings

4 K. Kahloot and P. Ekler, “Capturing Data Topology Using Graph-based Association Mining”, International Conference on Cloud Computing, GRIDs, and Virtual-

National Workshop Proceedings


6 References


