Spring 2020

Applying Artificial Intelligence to Medical Data

Shaikh Shiam Rahman

APPLYING ARTIFICIAL INTELLIGENCE TO MEDICAL DATA

by

SHAIKH SHIAM RAHMAN

(Under the Direction of Hayden Wimmer)

ABSTRACT

Machine learning, data mining, and deep learning has become the methodology of choice for analyzing medical data and images. In this study, we implemented three different machine learning techniques to medical data and image analysis. Our first study was to implement different log base entropy for a decision tree algorithm. Our results suggested that using a higher log base for the dataset, with mostly categorical attributes, with three or more categories for each attribute, can obtain a higher accuracy. For the second study, we analyzed mental health data and tuning the parameters of the decision tree (splitting method, depth, and entropy). Our results identified the most crucial attributes for the dataset. The final study is on the Kimia Path24 image dataset. We built and trained a deep convolutional neural network and tested different hypotheses of batch size, number of epochs, and learning rate. For the final study, all the hypotheses were supported by our experimental results.

INDEX WORDS: Machine learning, Deep learning, Medical data, Data science, Decision tree

APPLYING ARTIFICIAL INTELLIGENCE TO MEDICAL DATA

by

SHAIKH SHIAM RAHMAN

B.S., Khulna University of Engineering & Technology, Bangladesh, 2014

M.S., Georgia Southern University, 2020

A Thesis Submitted to the Graduate Faculty of Georgia Southern University in Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

APPLYING ARTIFICIAL INTELLIGENCE TO MEDICAL DATA

by

SHAIKH SHIAM RAHMAN

Major Professor:	Hayden Wimmer
Committee:	Lei Chen
	Cheryl Aasheim

Electronic Version Approved:
May 2020

DEDICATION

To God, for his grace, mercy, protection and provision all through my master's program. I am also grateful to my family, all faculty members of the Georgia Southern IT department, and my project supervisor for all the spiritual, moral, financial and intellectual support.

ACKNOWLEDGMENTS

A thesis cannot be completed without the help of many people. The author would like to acknowledge the efforts of my committee professors, Dr. Hayden Wimmer, Dr. Lei Chen, Dr. Cheryl Aasheim, and to my fellow graduate students.

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

LIST OF EQUATIONS

Page

LIST OF ALGORITHMS

Page

CHAPTER 1

INTRODUCTION

The classification approach of a decision tree is to measure the impurity of an attribute and split the data based on the categories of that attribute. The standard statistical method for measuring impurity is the logarithmic entropy calculation. This paper compares the accuracy for different logarithmic entropy functions on categorical datasets via the C4.5 decision tree algorithm. We implemented ln, log_{10} and log_2 logarithmic entropy equation and tested the differences of various categorical datasets. We found that if the dataset has mostly categorical attributes and most of those attributes have three or more class categories, then log_{10} and ln outperform log_2. This work demonstrates that one should use log_{10} or ln if the dataset has mostly categorical attributes and most attributes have three or more classification elements. When the dataset has mostly categorical attributes and most of the attributes have two classification elements, then the default log_2 entropy equation performs best.

Data mining lies at the interface of statistics, pattern recognition, and machine learning. An organized collection of data and proper data visualization are the main prerequisites of data mining. Proper use of data mining techniques will help identify important patterns and relationships in a dataset. In this work, we implemented a data mining algorithm on mental health data and found the most important attributes that trigger issues with mental health treatment. For this study, we used Microsoft Excel for data preparation and filtering, Microsoft SQL server as the data storage, and Microsoft SQL Server Analysis Service (SSAS) for building the data mining model. This is an important process which can help organizations provide a comfortable environment for employees that are facing issues with mental health treatment.

Deep learning methods recently made notable advances in the tasks of classification and representation learning. We built a deep convolutional neural network architecture and trained it with a Kimia Path24 dataset. This dataset has around 22.5K images for training and 1.3K images for testing with 24 different categories. In this study, we developed three different hypotheses and trained the network multiple times, changing the parameters of the learning network. We observed the effect of three parameters on our network architecture: 1) learning rate, 2) batch size, and 3) number of epochs. The results established the credibility of the hypotheses and showed that we can build networks with high accuracy by tuning the deep learning parameters based on the dataset. Our study suggests that deep convolutional neural networks can be useful to improve the accuracy of pathological images.

The rest of the thesis is organized into six chapters. The second chapter contains the literature review for three studies. In chapter three, we present our first study, "Improving the Predictive Performance of the C4.5 Decision Tree Algorithm for Categorical Medical Data." Here, we present the background of our study, the methods, and the comparison of our improvement with regular methods. In chapter four, we present our second study, "An Application of Data Mining of Mental Health Data." This study explains why mental health is important and how mental health disorders can be detected using a decision tree by tuning parameters. Our study of deep learning with Kimia Path24 dataset is embedded in chapter five. Here, we present the background of deep learning, why deep learning is useful for image processing, the architecture of our deep learning algorithms, our hypotheses regarding this dataset, and the support for those hypotheses. In chapter six, we summarize the whole thesis and the findings of all our studies regarding the medical data.

CHAPTER 2

LITERATURE REVIEW

Study 1 – Improving the Predictive Performance of the C4.5 Decision Tree Algorithm for Categorical Medical Data

Qin, Xia, Prabhakar, and Tu (2009) propose a new rule-based classification and prediction algorithm called uRule for classifying uncertain data. The authors also extend the rule for pruning, generating, and optimizing tree. The authors proposed a new measure called probabilistic information gain for generating rules and set a new probability distribution function, which will create a set of rules for calculation. This algorithm determined which rule is the best fit for a certain value(s).

Ludwig, Jakobovic, and Picek (2015) investigated a fuzzy decision tree (FDT) algorithm applied to the classification of gene expression data. The authors experimented with a fuzzy decision tree algorithm with a goal of analyzing gene expression cancer data. Besides the comparison with a decision tree algorithm, they also compared the proposed algorithm with several other well-known algorithms for classification. Both full and reduced feature sets were run with common data mining algorithms. The support vector machine algorithm outperformed all other data mining algorithms achieving 100% accuracy on some data sets.

W. Liu, Chawla, Cieslak, and Chawla (2010) introduced a new measure, Class Confidence Proportion (CCP). In decision trees, results in rules are biased towards the majority class. The authors designed CCP to overcome this bias. The authors introduced the CCP in information gain and used the improvised measure to construct the decision tree. The results compared the CCP-CART and CCP-C4.5 with base C4.5 and CART. CCP-CART and CCP-C4.5 outperformed the base C4.5 and CART.

Yana, Maa, Zhaob, and Kokogiannakis (2016) presented a decision tree-based data-driven diagnostic strategy for an air handling unit (AHU). Decision tree, a well-known classifier, has been applied in the prediction of building energy usage with satisfying accuracy. The interpretability of the proposed strategy can be helpful in understanding the diagnostic strategy. Data-driven methods are superior in extracting useful information from large data sets and modeling. Combining data-driven methods with expert knowledge might be a possible solution for developing effective data-driven based fault diagnostic strategies (Yana et al., 2016).

Rutkowski, Jaworski, Pietruczuk, and Duda (2014) proposed a new algorithm for mining streamed data. The new algorithm is based on CART, the modification is called dsCART. To solve the attribute splitting problem, the authors applied the Gaussian distribution algorithm. For testing the accuracy, the algorithm had been tested with Syntactic and real data. The accuracy and other parameters had been compared with McDiarmid and Gaussian decision tree algorithm. Results showed that this is a proper tool for solving streamed data classification.

Study 2 – An Application of Data Mining of Mental Health Data

Gnanlet and Gilland (2009) conducted a full-factorial numerical experiment and found the benefit of using staffing decision under flexibility. This study discussed about 4 configurations under flexibilities: (1) no flexibility, (2) demand upgrades, (3) staffing flexibility, and (4) demand upgrades and staffing flexibility. This study helped hospital managers determine optimal staffing and capacity decision making. This experiment concluded that centralized decision making can yield a greater benefit than decentralized decision making.

Obenshain (2004) mentioned three different health care situations where data mining algorithms have been implemented successfully with satisfying outcomes: (1) association rules helped to enhance control over infection, (2) clustering and association rules were used to rank the

hospitals, and (3) American Healthways used predictive modeling technology to predict the likelihood of short-term health problems. This study compared statistics and data mining techniques and discussed their combination to develop the best practice.

Meyer et al. (2014) used a decision tree to improve the performance of a dynamic decision-making system. This study applied data mining classification techniques to the collected data for discovering the conditions which were different for dynamic decision-making strategies. This information was used to improve the decision-making strategies. This study used a predictive data mining technique (decision tree) to identify the failed conditions. The results suggested different data mining techniques (e.g., decision tree, neural network) are useful for the performance improvement of complex and ill-structured dynamic environments.

A. Gupta, Wilkerson, Sharda, and Colston (2018) developed a prediction model for the identification of college football player's injury risk. This study developed an injury risk assessment model that is based on external factors (e.g., exposure to the task, performance role) and intrinsic factors (e.g., injury history, movement efficiency). This study used logistic regression and Cox regression for assessing the injury of football players.

Study 3 – Deep Learning: An Imperial Study on Kimia Path24 Dataset

Deep Learning

Deep learning-based image super-resolution (SR) models have been actively explored and often achieve the state-of-the-art performance on various benchmarks of SR. A variety of deep learning methods have been applied to tackle SR tasks ranging from the early Convolutional Neural Networks based method to recent promising SR approaches using Generative Adversarial Nets (Wang, Chen, Hoi, & Intelligence, 2020). Dong, Loy, He, and Tang (2014) proposed a deep learning method for a single SR. In this study, the authors contributed three aspects: 1) presented

a convolutional neural network for image super-resolution, 2) established a relationship between the deep learning-based SR method and the traditional sparse coding-based SR methods, and 3) demonstrated the usefulness of deep learning in the classical computer vision problem with super-resolution. The method of this study directly established an end-to-end mapping between the low/high-resolution images. The mapping is represented as a deep convolutional neural network (CNN) that takes the low-resolution image as the input and outputs the high-resolution. With a lightweight structure, the Super-Resolution Convolutional Neural Network (SRCNN) has achieved a superior performance above other state-of-the-art methods. The authors conjectured that an additional performance could be further gained by exploring more hidden layers/filters in the network using different training strategies (Dong et al., 2014).

Y. Chen, Lin, Zhao, Wang, and Gu (2014) introduced the concept of deep learning into hyperspectral data classification. They introduced the deep learning-based feature extraction for hyperspectral data classification. Their method focused on applying an autoencoder (AE). At first, the authors verified the eligibility of stacked autoencoders by following classical spectral information-based classification. After that, a new way of classifying with spatial-dominated information was presented. Finally, they proposed a deep learning framework by merging the two features. Y. Chen et al. (2014) exploited a single layer autoencoder (AE) and a multi-layer stacked AE (SAE) to learn shallow and deep features of hyperspectral data. AE-extracted features are useful for classification. AE and SAE deep feature extraction models increased the accuracy of SVM and logistic regression while obtaining the highest accuracy when compared with other feature extraction methods (Y. Chen et al., 2014).

Chetlur et al. (2014) created a library with optimized routines for deep learning workloads with a similar intent to Basic Linear Algebra Subroutines (BLAS) (BLAS, 2014). They presented

a novel implementation of convolutions that provided a reliable performance across a wide range of input sizes, and they took advantage of the highly-optimized matrix multiplication routines to provide a high performance without requiring any auxiliary memory. Integrating Nvidia CUDA Deep Neural Network's (cuDNN) library into Caffe improved the performance by 36% on a standard model with a reduced memory consumption. NVIDIA cuDNN's performance is 86% of the maximum performance, with a small mini-batch size of 16. This implementation performed well across the convolution parameter space. NVIDIA cuDNN's library ranged from 23-35% of peak performance on the Tesla K40 and from 30-51% of peak performance on the GTX 980. NVIDIA cuDNN's library provided a performance portability across GPU architectures with no need for users to retune their code as GPU architectures evolve (Chetlur et al., 2014).

A large body of the work in deep learning can be classified into: (1) generative, (2) discriminative, and (3) hybrid categories (Deng, 2014). A deep autoencoder is used for learning efficient encoding or dimensionality reduction for a set of data. It is a non-linear feature extraction method classified as generative. Deep architecture, consisting of both pretraining and fine-tuning stages in its parameter learning, is classified as a hybrid. The concept of stacking, where simple modules of functions are composed first and then they are "stacked" on top of each other in order to learn complex functions, is classified as discriminative (Deng, 2014).

Algorithm-level noise tolerance can be leveraged to simplify underlying hardware requirements. Noise tolerance can lead to a co-optimized system that achieves significant improvements in computational performance and energy efficiency. Deep networks can be trained using only 16-bit wide fixed-point number representation using stochastic rounding with little degradation in classification accuracy (S. Gupta, Agrawal, Gopalakrishnan, & Narayanan, 2015).

Deep learning methods have dramatically improved the state-of-the-art speech recognition, visual object recognition, object detection, and many other domains, such as drug discovery and genomics (LeCun, Bengio, & Hinton, 2015). Deep convolutional nets have brought about breakthroughs in processing images, video, speech, and audio; whereas, recurrent nets have shined the light on sequential data, such as text and speech. Deep learning has beaten other machine-learning techniques by predicting the activity of potential drug molecules, analyzing particle accelerator data reconstructing brain circuits, and determining the effects of mutations in non-coding DNA on gene expression and disease. The 2012 ImageNet competition success has brought about a revolution in computer vision. ConvNets are now the dominant approach for almost all recognition and detection tasks. ConvNets also enhanced human performance on some tasks. The combination of ConvNets and the recurrent net modules generate stunning demonstrative image captions (LeCun et al., 2015).

Cross modality feature learning can achieve a better feature for one modality (e.g., video) if multiple modalities (e.g., audio and video) are present at the feature learning time (Ngiam et al., 2011). Ngiam et al. (2011) presented a series of tasks for multimodal learning showing how to train deep networks to learn features that address these tasks. These models were validated on the CUAVE and AVLetters datasets on audio-visual speech classification, demonstrating the best published visual speech classification on AVLetters and effective shared representation learning. Learning a canonical correlation analysis (CCA) with a shared representation of raw data results in a good performance. Learning the CCA representation on the first layer feature results in a significantly better performance compared to the original modalities for supervised classification (Ngiam et al., 2011).

Papernot et al. (2016) formalized the space of adversaries against deep neural networks (DNNs) and introduced a novel class of algorithms to craft adversarial samples based on a precise understanding of the mapping between inputs and outputs of DNNs. This experiment formally described a class of algorithms for crafting adversarial samples misclassified by DNNs using three tools: the forward derivative, adversarial saliency maps, and the crafting algorithm. These tools were applied to a DNN and used for a computer vision classification task: handwritten digit recognition. The crafting algorithm can reliably produce samples correctly classified by human subjects but misclassified in specific targets by a DNN with a 97% adversarial success rate while only modifying on average 4.02% of the input features per sample (Papernot et al., 2016).

Deep Learning in Medical Images

Brosch and Tam (2013) described a novel method called multi-scale structured convolutional neural networks (MS-CNN) for learning the manifold of 3D brain images. The method does not require the manifold space to be locally linear, and it does not require a predefined similarity measure or a prebuilt proximity graph. This manifold learning method was based on deep learning, a machine learning approach that uses layered networks (called deep belief networks, or DBNs). The authors proposed a computationally efficient training method for DBN. The proposed method performed manifold learning by reducing the dimensionality of the input images using a DBN. This method used deep learning to discover patterns of similarity and variability within a group of images. The learned manifold coordinates captured shape variations of the brain that correlated with demographic and disease parameters. The MS-CNN algorithm was much more efficient than traditional, convolution-based methods (Brosch & Tam, 2013).

Plis et al. (2014) used deep learning to analyze the effect of parameter choices on data transformations. The authors demonstrated their results in the application of deep learning methods

to structural and functional brain imaging data. They also described a novel constraint-based approach to visualize high dimensional data. These methods included deep belief networks and the building block of the restricted Boltzmann machine. This study presented recent advances in the application of deep learning methods to functional and structural magnetic resonance imaging. The main goal was to validate the feasibility of this application by: (1) investigating if a building block of deep generative models a restricted boltzmann machine, (2) examining the effect of the depth in deep learning analysis of structural magnetic resonance imaging (MRI) data, and (3) determining the value of the methods for discovering the latent structure of a large-scale. Deep learning has a high potential in neuroimaging applications. The depth of the DBN helped classification and increased group separation. DBNs have a high potential for exploratory analysis (Plis et al., 2014).

A. Payan and G. J. a. p. a. Montana (2015) used deep learning methods, spars autoencoders, and 3D convolutional neural networks to build an algorithm that can predict the disease status of a patient based on an MRI scan of the brain. This study demonstrated that the 3D convolutional neural networks outperformed several other classifiers. The authors compared the performance of 2D and 3D convolutional networks. This study took a two-stage approach. The authors used a sparse autoencoder to learn filters for convolution operations, and then built a convolutional neural network in which the first layer uses the filters with the autoencoder. The 3D approach had a superior performance for the 3-way comparison. The 3D approach had the potential to capture local 3D patterns, which may boost the classification performance, albeit only by a small margin (A. Payan & G. J. a. p. a. Montana, 2015).

Recently, deep learning methods introduced a medical image analysis with promising results in multiple applications, including computerized prognosis for Alzheimer's disease (S. Liu

et al., 2014), tumor segmentation (Havaei et al., 2017), and histopathological diagnosis (Cireşan, Giusti, Gambardella, & Schmidhuber, 2013; Litjens et al., 2016). Neuroimaging analysis is frequently utilized for increasing diagnostic ability; however, some studies are using deep learning models to discover the diverse patterns in patient data characteristics of a disease (Brosch, Yoo, Li, Traboulsee, & Tam, 2014; Kim, Calhoun, Shim, & Lee, 2016; Suk, Lee, Shen, & Initiative, 2014; Suk, Wee, Lee, & Shen, 2016). The implementation was done for this classifier using a deep neural network initialized by a DBN (DBN-DNN).

Pinaya et al. (2016) trained a deep learning model, known as DBN, to extract features from brain morphometry data. The deep learning models excel at neuroimaging-based prediction methods and can be useful for demonstrating complex and subtle associations, as well as enabling more accurate individual-level clinical assessments. The strength of deep architecture came from multiple levels of non-linear processing that are well-suited to capture highly varying functions with a compact set of parameters. The deep architecture provided superior performance in classification tasks. The DBN highlighted differences between classes, especially in the frontal, temporal, parietal, and insular cortices and in some subcortical regions, including the corpus callosum, putamen, and cerebellum (Pinaya et al., 2016).

Bao and Chung (2018) stated that specific architecture is designed to capture discriminative features for each sub-cortical structure. To improve the performance of CNN, two intuitive ways are generally utilized. Apart from the straightforward enlargement of network architecture, some elegant micro-structures have also been designed to enhance the capability recently. To evaluate the performance of the proposed method, the comparison with other methods has been carried out. As several different voxel resolutions exist in each dataset, affine transformations between atlases and the target image were first conducted as pre-processing. The initial structural surface used in

label consistency (Blue Curve at Iteration 0) was generated by fusing the warped label maps with majority voting (MV). The results of MV was also employed at a baseline for comparison. The results indicated that, with a multi-scale strategy, more discriminative features can be captured, and the labeling result can be improved. Due to the lack of constraints among testing patches, embracing learning method alone often led to a rough boundary and desultory segmentation result. Experimental results demonstrated that the proposed method can obtain a better performance as compared to other state-of-the-art methods (Bao & Chung, 2018).

de Brebisson and Montana (2015) proposed a methodology based on a deep artificial neural network that assigned each voxel in an MR image of the brain to its corresponding anatomical region. The inputs of the network captured information at different scales around the voxel of interest: 3D and orthogonal 2D intensity patches captured a local spatial context while downscaling large 2D orthogonal patches and distances to the regional centroids enforce global spatial consistency. The combined use of three 2D orthogonal patches dramatically improved the segmentation performance compared to 2D or 3D patches. The distances to centroids, in addition to their invariance qualities, significantly outperformed the coordinates. For already manually segmented brains, using estimated centroids yield equivalent results as using the true centroids. It has a mean dice coefficient of 0.725 and an error rate of 0.163 when evaluated on the 20 testing MRIs of the MICCAI challenge. Good validation results were obtained by training huge networks, sometimes composed of tens of millions of parameters, with a relatively small amount of data. The trained networks overfit the training data; however, they still generalize fairly well to unseen MRIs (de Brebisson & Montana, 2015).

H. Chen, Dou, Yu, and Heng (2016) explored the deep residual learning on the task of volumetric brain segmentation. First, they proposed a deep voxelwise residual network, referred to as VoxResNet. Second, an auto-context version of VoxResNet is proposed by seamlessly integrating the low-level image appearance features, implicit shape information, and high-level context together for improving the volumetric segmentation performance. The results of combining multi-modality and auto-context information give more accurate results visually than only multi-modality information. The proposed algorithm has an application beyond brain segmentation, and it can be applied in other volumetric image segmentation problems (H. Chen et al., 2016).

Choi and Jin (2016) developed a fast and accurate method for the striatum segmentation using deep convolutional neural networks (CNN). The delicate segmentation process performed by Local CNN used small portions of the image rather than the whole image. This method suggested that FreeSurfer segmentation included slightly more true-positive voxels and much more false-positive voxels than the CNN-based approach, which resulted in lower Dice Similarity Coefficient (DSC). This approach depended on training using manual segmentation, another recently developed automatic segmentation tool based on multimodal images (MIST), did not require a manually labeled training set, which could be flexible for various types of data (Choi & Jin, 2016).

CHAPTER 3

IMPROVING THE PREDICTIVE PERFORMANCE OF THE C4.5 DECISION TREE ALGORITHM FOR CATEGORICAL MEDICAL DATA

Introduction

Decision tree is most popular rule-based classification algorithm. Decision tree is a decision support tool that uses a tree-like model of decision and their possible consequences and future event outcomes. Entropy is the most critical section of the decision tree, which is used for calculating the impurity of an attribute. The success of a decision tree depends on the assumption that every attribute in the training data set has an equal number of class instances evenly distributed among them. If the attributes have binary classification and there is similar number of instances for each class, then C4.5 performs well. When the training dataset contains mostly categorical attributes and the classification of those attributes is not binary or contains three or more classification elements, using a higher log base for calculating impurity can outperform the C4.5 algorithm.

The objective of this method is to discover the effect of higher log base over the attributes with three or more classification elements. For this purpose, several features have to be kept in mind. The dataset has to contain mostly categorical attributes. The more categorical attributes with three or more classification elements, the higher the chance is, that the higher log base will outperform the basic C4.5.

Methods

C4.5 was introduced by Ross Quinlan in 1993. Algorithm 3.1 shows the pseudocode of C4.5 algorithm (Quinlan, 1993). Algorithms for constructing decision trees are among the most well-known and widely used of all machine learning methods. Among decision tree algorithms, J.

Ross Quinlan's ID3 and its successor, C4.5, are probably the most popular in the machine learning community (Salzberg, 1994). C4.5 has its origins in Hunt's concept learning systems by way of ID3 (Quinlan, 1993). C4.5 and its predecessor, ID3, use formulas based on information theory to evaluate the "goodness" of a test; in particular, they choose the test that extracts the maximum amount of information from a set of cases, given the constraint that only one attribute will be tested (Salzberg, 1994).

C5.0 is an improvement on C4.5 which is commercially sold. C5.0 is a more advanced version of Quinlan's C4.5 classification model that has additional features, such as boosting and unequal costs for different types of errors. Like C4.5, it has tree-based and rule-based versions and shares much of its core algorithms with its predecessor (Kuhn & Johnson, 2013).

This algorithm constructs the decision tree with a divide and conquer strategy. For constructing an effective decision tree, dataset must fulfill the following key requirements: 1) attribute value description, 2) predefined classes, 3) discrete classes, 4) sufficient data, and 5) logical classification model. In the beginning only the root is present with the whole training set. Then for each excluded attributes and instances, the C4.5 calculates the gain ratio for each attribute.

```
calculateDecisiotree(TS) {
  compute attributes info gain
  if(oneClass or no info gain ) {
    mark the caller child as a leaf node
    return
  }

  for each attribute{
    if (attribute is discreate){
      calculateinfo gain ratio
    } else {
      calculate threshold
    }
  }

  Get the best attribute.
  split the node among the number of classes the attribute have
  exclude the attribute from attribute list
  for child node {
    child[i] = calculateDecisionTree(TS-exclude current attribute)
  }
}
```

Algorithm 3.1: Quinlan's C4.5 Decision Tree Algorithm

Here, in this work, we propose a variation of the logarithmic calculation Algorithm 3.2 to calculate the entropy equation (3.1) and measure the accuracy and performance of different categorical datasets. A small modification in logarithmic calculation can make an impact on accuracy and performance. As most of the categorical datasets are not split on binary characteristics. A small modification based on the classification of attributes can make an impact.

Global variable log Base Number as short integer
Log Base Number = e, 2, 10

$$Entropy(S) = -\sum_{j=1}^{k} \frac{freq(C_j, S)}{|S|} * log_{base}(\frac{freq(C_j, S)}{|S|}) \quad (3.1)$$

In our algorithm, we propose to use the variable log base for calculating the entropy.

```
calculateDiscreteEntropy(Target, TS) {
    // return the entropy of the Target attribute
    C = getDiscreateClassValues(Ts, Target)
    Entropy = 0
    for each Class in C do
        ClassFrequency = FrequencyOfClass(Ts,Ci)
        ClassRatio = ClassFrequency / sizeOf(TS)
        LogValue = LogBase(ClassRatio,base)
        Entropy += (- ClassRatio * LogValue)
    end for
    return Entropy
}
```

Algorithm 3.2: log4.5 Entropy Calculation.

Results

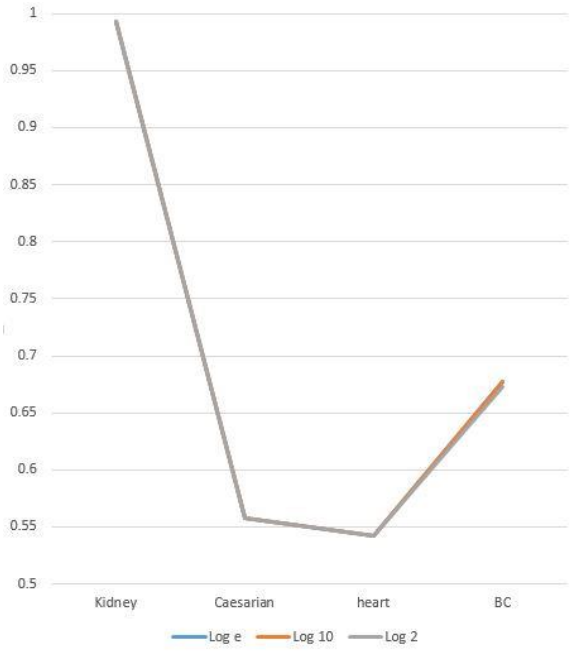

Figure 3.1: The Comparison Between All Datasets.

We test the model with University of California Irvine (UCI) machine learning dataset. We used 4 different datasets to measure the performance of our algorithm. From Figure 3.1 we can see that the log_{10} outperforms log_2 with accuracy 0.677 for the breast cancer dataset because this dataset contains 10 attributes and all of them are categorical. Among them 5 attributes have 3 or

more classification elements. For rest of the datasets, changing of the log base has no effect on the accuracy.

When a mixture of continuous and categorical attributes change the log base of the C4.5 decision tree algorithm has no impact on performance. Conversely, when all attributes are categorical, changing the log base of C4.5 can have an impact on performance as defined by classification accuracy. From the above discussion, we can say that the log_{10} will outperform log_2if most of the attributes are categorical and the categorical attributes have 3 or more classification elements.

Conclusion

This work investigated log-based entropy calculations for decision tree implementation. The decision tree was tested on 4 different UCI machine learning datasets. Four datasets were analyzed and compared with log_2 base entropy calculated decision tree algorithm. Higher values of accuracy were achieved on log_{10}, when the dataset contains mostly categorical attributes and those attributes have 3 or more classification elements (breast cancer). This reveals that higher accuracy is achieved for log_{10} when most of the attributes are categorical with 3 or more classification elements.

CHAPTER 4

AN APPLICATION OF DATA MINING OF MENTAL HEALTH DATA

Introduction

The purpose of this project is to apply data mining techniques to predict whether an employee in a tech company has asked for mental health treatment or not. We aimed to identify and disclose very important and useful information that will be beneficial and effective for the tech organizations.

To identify opportunities for mental health improvement, we planned to identify the best attributes that are responsible for mental health issues. We used data mining algorithms to find the attributes that are responsible for mental health issues.

Methods

For building the decision tree with this dataset, we have used SQL Server Analysis Services (SSAS). Data partitioning is an important part of preparing the mining model and testing the mining model. We partitioned our data into two subsets: (1) training data and (2) testing data. Training data is used to build the tree, while testing data is used to test the validity of the model. The recommended ratio of training and test instances for a given dataset was 20:1 (Larose & Larose, 2015). After partitioning, we have a total of 1,172 instances for building the tree and 61 instances to test the model.

SSAS comes with various built-in features, like using different data sources for input and output, building the mining model based on different parameters related to that mining model. SSAS also suggests the attributes which are the best fit to predict the outcome. SSAS provides an option for choosing the ratio of training and test data as well as tools to validate the model such, as classification matrices and lift charts.

Our plan was to build different models based on three SSAS decision tree building parameters: complexity penalty, score method, split method. In the beginning, all the models were built with the parameters' default value. Then we have changed the parameters' value to different ranges and built different models based on those changes.

Complexity Penalty: Inhibits the growth of the decision tree. Decreasing the value increases the likelihood of a split, while increasing the value decreases the likelihood. The default value is based on the number of attributes for a given model: the default is 0.5 if there are 1 to 9 attributes: the default is 0.9 if there are 10 to 99 attributes; and the default is 0.99 if there are 100 or more attributes (Services, 2018).

Score Method: Specifies the method used to calculate the split score. The available methods are: Entropy (1), Bayesian with K2 Prior (3), or Bayesian Dirichlet Equivalent with Uniform prior (4) (Services, 2018).

Split Method: Specifies the method used to split the node. The available methods are: Binary (1), Complete (2), or Both (3) (Services, 2018).

Results

The change of accuracy with respect to parameter values is displayed in Table 4.1. The list of attributes that are useful for building the tree has been chosen in three steps for each parameter. First step, we used those attributes that were suggested by the Microsoft SQL Server Analysis Services. Then, those attributes that have relevance more than 0.1 and finally all the attributes. Few parameter values didn't build a decision tree. Those accuracies are left blank. Work interference and the family history were identified as the most important attributes for predicting the mental health treatment, from Figure 4.1.

#	Parameter value	Dataset Attributes	Accuracy
1.	Complexity penalty is set to default choice	Suggested	80.32
2.		Relevant	75.41
3.		All	77.04
4.	Complexity penalty = 0.5	Suggested	81.96
5.		Relevant	78.69
6.		All	--
7.	Complexity penalty = 0.9	Suggested	88.52
8.		Relevant	--
9.		All	80.32
10.	Complexity penalty = 0.99	Suggested	83.6
11.		Relevant	85.24
12.		All	75.41
13.	Score Method = 1	Suggested	75.4
14.		Relevant	86.6
15.		All	67.21
16.	Score Method = 3	Suggested	--
17.		Relevant	80.32
18.		All	81.96
19.	Split Method = 1	Suggested	68.85
20.		Relevant	77.04

21.		All	83.6
22.	Split Method = 1	Suggested	--
23.		Relevant	86.88
24.		All	--

Table 4.1: Decision Tree Accuracy with Different Parameter Values.

Among 61 test cases, a total of 53 cases were identified correctly with 8 incorrect predictions. The model has 88.5% accuracy.

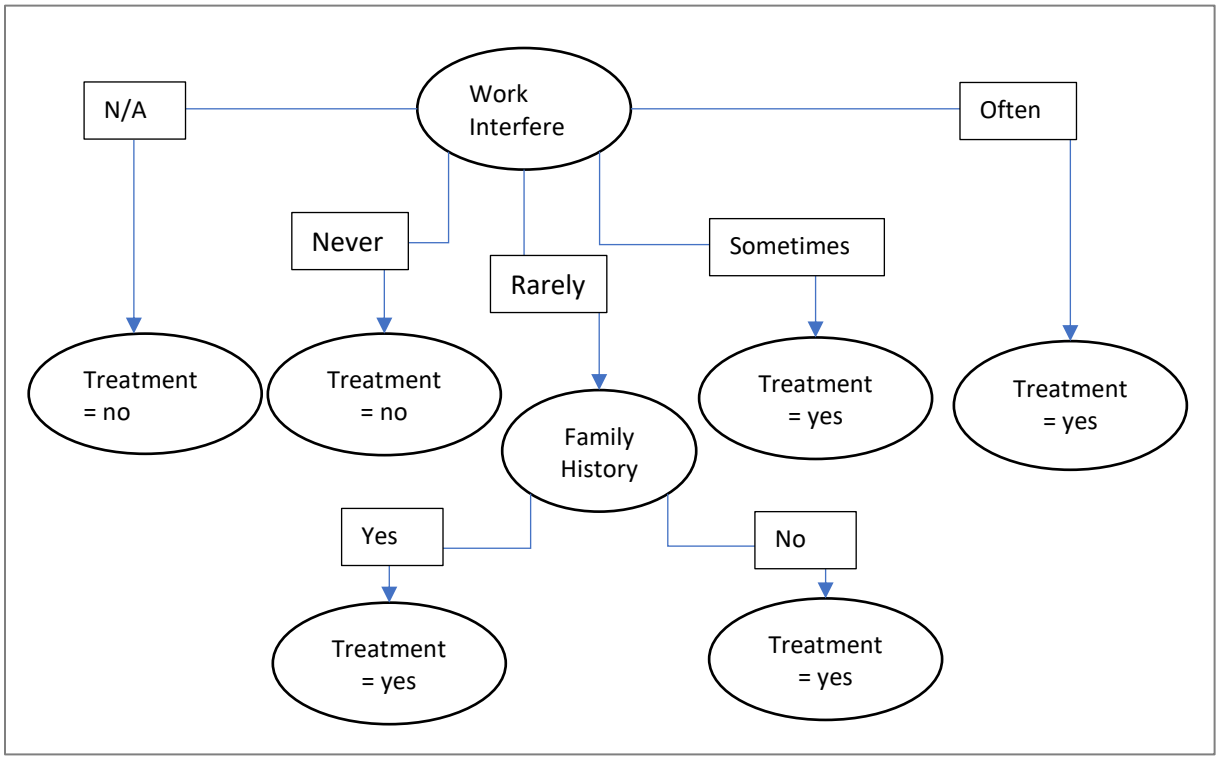

Figure 4.1: SSAS Decision Tree with Highest Accuracy.

Conclusion

The objective of this project was to develop a detective system that can assist tech companies to help them identify their employees' mental health issues. This will help the tech

companies to identify whether their employees will ask for the mental health support. 24 different decision trees were used with different parameter values. The work interference and the family history were counted as the most important attributes for predicting the mental health issue.

From the model, Figure 4.1, it is clear that the employees with most work interference has the highest probability to ask for mental health treatment. The family history also plays an important role to provoke the mental health issue and ask for treatment.

CHAPTER 5

DEEP LEARNING: AN IMPERIAL STUDY ON KIMIA PATH24 DATASET

Introduction

For several decades, pathology has been described as the archiving of microscopic information of specimens. This microscopic information is organized by storing specimens on glass slides. The problem with glass slides is that they are fragile and require a very large specially prepared storage room to store the specimens in. This kind of storage requires a lot of logistical infrastructures.

In 1999, whole slide imaging (WSI) was introduced by Wetzel and Gilbertson (Ho et al., 2006). WSI can provide high image quality that doesn't decay over time, along with a range of other benefits. WSI can be used by multiple researchers to investigate multiple slides at the same time, and this kind of data is more useful in order to retrieve information and maintain quality control.

Pathology bounded by the WSI system is emerging into an era of digital specialty. WSI is providing solutions for centralizing diagnostic by improving the quality of diagnosis, patient safety, and economic concerns. (Ghaznavi, Evans, Madabhushi, & Feldman, 2013).

The diagnosis of WSI is still difficult. The gigapixel nature of WSI scans makes it difficult to store, transfer, and process samples in real-time. One also needs tremendous digital storage to archive them (Babaie et al., 2017). A diagnostic system aided by digital pathology scanned data would allow for a more objective approach, and increase our ability to predict an individual's pathological diagnosis and treatment response.

The most common form of machine learning is supervised learning. In computer vision, deep convolutional networks have now become the technique of choice (Litjens et al., 2017). Deep

learning is making major advances in solving problems in the artificial intelligence community. Deep learning deals with the problem of data representation by introducing simpler intermediate representations that allow them to combine in order to build complex concepts (Affonso, Rossi, Vieira, & de Leon Ferreira, 2017). Deep learning methods are multiple levels of representation. These representations are obtained by composing simple but non-linear modules that transform the representation at one level into a representation at a higher abstract level. Recent studies have shown that machine learning algorithms were able to predict disease more accurately than experienced clinicians (A. Payan & G. Montana, 2015). Deep learning has a large interest in medical image analysis. It is expected that deep learning will hold $300 million for the medical imaging market by 2021 (Razzak, Naz, & Zaib, 2018). It is of great interest to develop and improve such prediction methods.

Background

Single Neural Network

The basic unit of every single network is the neuron. This basic unit is also known as a node. The main computational principle of a neural network is that it receives inputs from any external sources and generates an output.

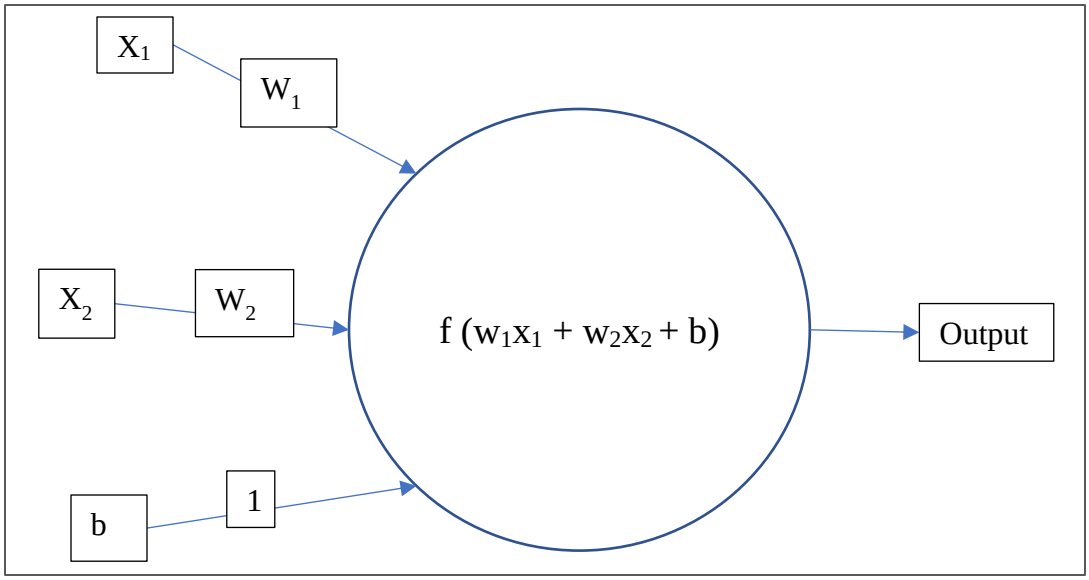

Figure 5.1: Single Neuron.

Each input has an associated weight which is assigned based on its source, as shown in Figure 5.1. The node applies a function to the weighted sum of the input. The function that is used for calculating output is called the activation function. The above network, shown in Figure 5.1, contains 2 input nodes X_1 and X_2. The associated weights are W_1 and W_2. There is another input with weight 1 and value b, which is known as bias.

The output of this neuron is calculated as shown in Figure 5.1. The function we use to calculate the value of the neural network is known as the activation function. This function is a non-linear function, which is used for introducing the non-linearity to the output of the neural network. According to most of the real-world dataset, the inputs are discrete and non-linear, and our main target is to train our network with this non-linear representation.

Every activation function takes a single number and performs a specific fixed mathematical operation (Karpathy, 2019). There are several activation functions that can be used based on the input of the neural network.

Feed forward neural network

The feed forward neural network was the first and simplest type of artificial neural network devised (Schmidhuber, 2015). This network model contains multiple basic units referred to as nodes arranged in layers, followed by the same type of adjacent layer. Nodes from adjacent layers have a connection between them, but nodes from the same layer have no connection among them. These connections are called edges. Each adjacent edge has an associated weight with it.

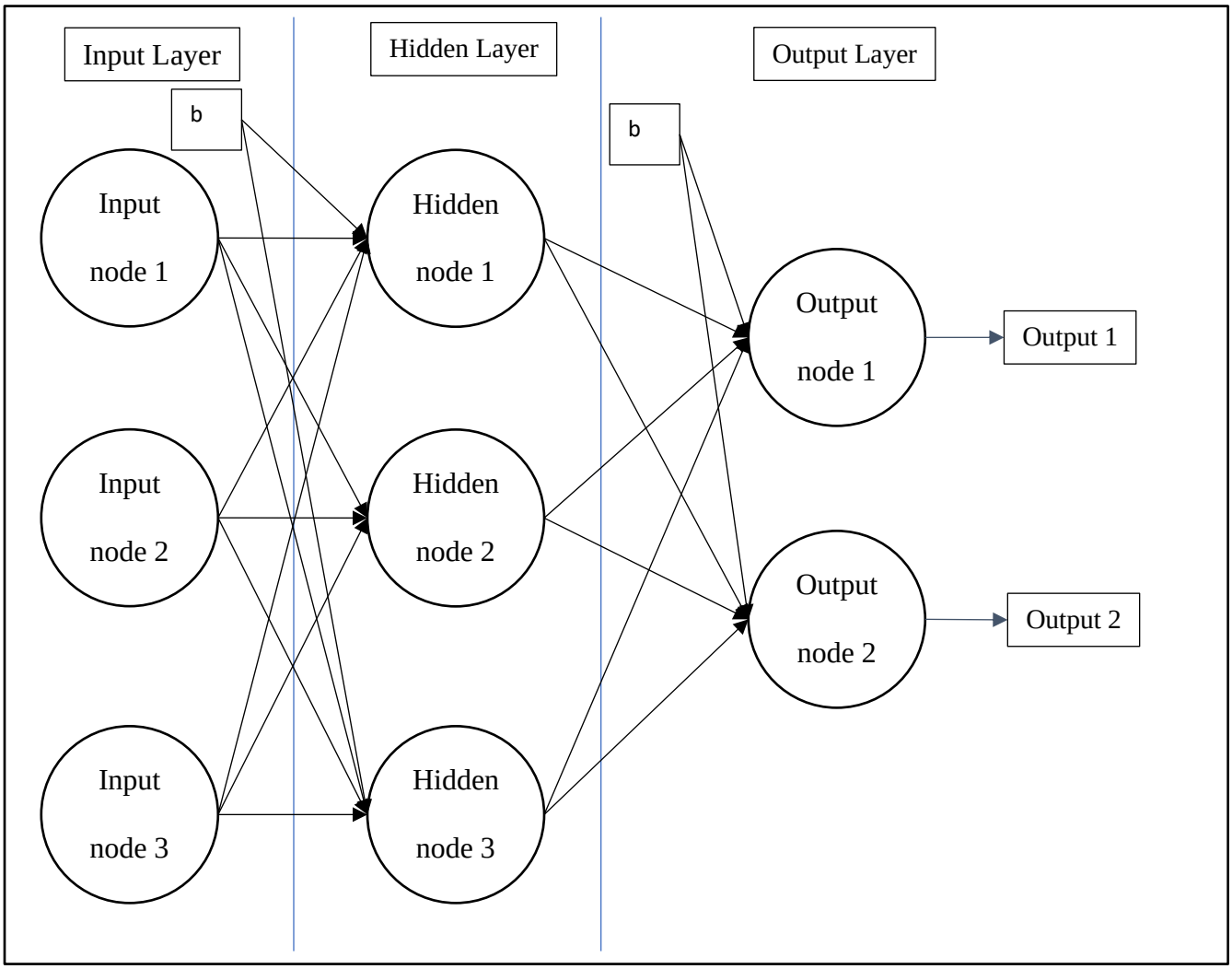

Figure 5.2: Feed Forward Neural Network.

There are three types of node layers in a feed forward neural network. These are: 1) input nodes, 2) hidden layers, and 3) output node. Three types of nodes in a network are shown in Figure 5.2.

1) **Input layer**: The input nodes gather the information from the outer world. The combination of input nodes is known as the input layer. These nodes are primarily responsible for receiving inputs from the dataset. No computations are performed in this layer.
2) **Hidden Layer**: Hidden layers are those that take inputs from the previous input layer. For each input layer, first, the input and weight of the connections are multiplied, and then all the input weight multiplication products are summed up. After that, the result of the summation is put through an activation function and the output is forwarded to the next layer. Equation (5.1) shows the output calculation for each node (Litjens et al., 2017). Here, the function in equation (5.1) is called an activation function.

$$\text{f(summation)} = f(w_0 * 1 + W_1 * X_1 + W_2 * X_2) \quad (5.1)$$

3) **Output Layer**: The output nodes also works the same way as the hidden nodes. These nodes take inputs from the previous hidden layer. For each output node, they first take the multiplication of the output of the previous node and the weight of the connection. Then, they sum up the multiplication results of all connections and put the summation on an activation function. The final step is to publish the outcome of the activation function as the output of the network.

Activation Function

Sigmoid: Equation (5.2) shows the sigmoid activation function. This takes a real value input and returns a value between 0 and 1.

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{5.2}$$

Tanh: Equation (5.3) shows the Tanh activation function. It takes a real value input and returns a value between -1 and 1

$$tanh\,(x) = 2\sigma(2x) - 1 \tag{5.3}$$

ReLU: ReLU stands for Rectified Linear Unit. It takes a real value input and thresholds it between the max and cuts the negative values to 0. Equation (5.4) shows the ReLU activation function.

$$f(x) = \max(0, x) \tag{5.4}$$

Softmax: The softmax activation function turns the numbers into possibilities that sum to one. Softmax activation function changes the outcomes to a vector, which represents the probability distribution of a list of potential outcomes. Equation (5.5) shows the sigmoid activation function (Litjens et al., 2017).

$$S(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{k} e^{x_j}} \tag{5.5}$$

Back Propagation

Calculating the Total Error

Once we have the result for each output node, our next goal is to calculate the error for each node. We can calculate the error for each output node using the squared error function and

sum them to get the total error. Equation (5.6) shows how we calculate the error for back propagation. Here, target is the expected value from the output node, and actual is the calculated result after forward propagation using the activation function.

Our goal with back propagation is to update each weight of the network connections, so they cause the actual output to be closer to the target output. This helps minimize the error of each node and the network as a whole.

$$Error_{total} = \sum \frac{1}{2}(target - actual)^2 \tag{5.6}$$

The Backwards Pass

To accomplish that goal, we have to calculate the derivative of the $Error_{total}$ with respect to the weights of the output layers. Let's consider w_i is the weight between the first hidden layer and the first output layer. If we want to know how much change in w_i will affect the total error, then our derivative will be $\frac{\partial E_{total}}{\partial w_i}$. The derivative of $Error_{total}$ with respect to the weight of the connection is called "the gradient with respect to w_i." From the $Error_{total}$ to w_i, there is no direct connection. We have to apply the chain rule to establish the relation among them. Equation (5.7) shows the chain rule of the partial derivative of $Error_{total}$ with respect to w_i.

$$\frac{\partial Error_{total}}{\partial w_i} = \frac{\partial Error_{total}}{\partial Output_1} * \frac{\partial Output_1}{\partial\, f(summation)} * \frac{\partial f(summation)}{\partial w_i} \tag{5.7}$$

Figure 5.3 shows how the weights are updated based on the total error. When all the new weight calculations are complete, then we update the weights as shown in equation (5.8). Here, η is known as the learning rate. This learning rate decides what portion of weight needs to be

updated. If the learning rate is high, then the weight changes very frequently and updates the decision. If too low, then the change requires a large number of training data.

$$w_{new} = w_{old} - \eta * \frac{\partial Error_{total}}{\partial w_i} \tag{5.8}$$

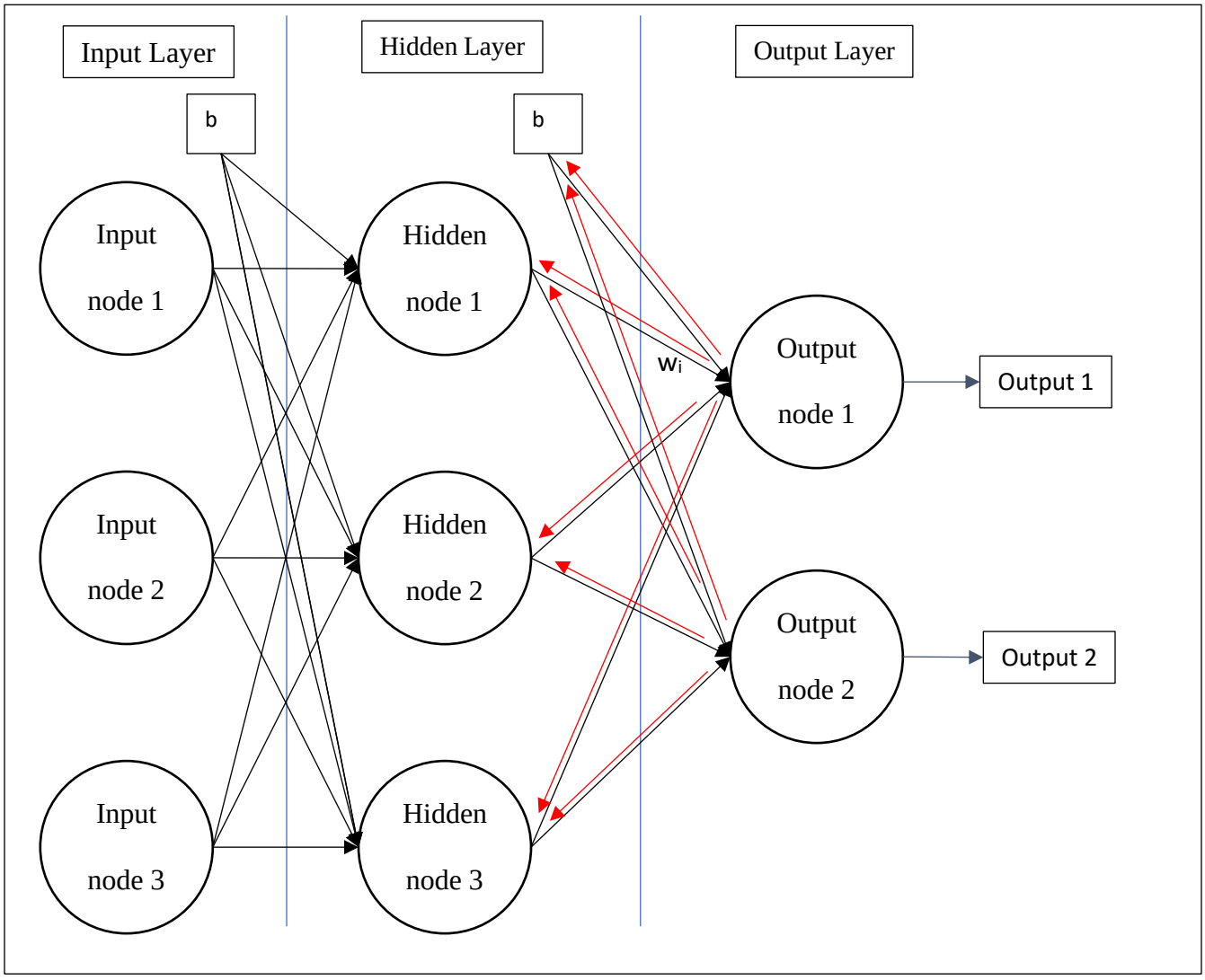

Figure 5.3: Back Propagation on Multiple Perception Neural Network.

Reducing Loss

Gradient Descent: Calculating the loss function for every single input is not a very efficient approach to find the convergence point. The convex problems have only one minimum data place where the slope is exactly 0. That is the minimum function where the loss function converges. The best approach to obtain the convergence point is called the gradient descent. The first step is to pick a random number between 0 and 1. The gradient descent algorithm then calculates the gradient of the loss curve at the starting point. Something that should be noted is that our gradient needs to have the property of moving both forward and backward. So, we pick our gradient as a vector, which has both direction and magnitude. The gradient always points towards the increase of the loss function. The gradient descent algorithm takes a step in the direction of the negative gradient in order to reduce loss as quickly as possible.

Stochastic Gradient Descent: In gradient descent, we don't use the whole dataset on a single iteration. We only use a certain amount of data at a single time, making it flow through the network and calculating the gradient. We call this a single iteration. When the dataset is large, then a single portion of data may take a long time to complete computation. A large dataset with randomly sampled examples probably contains redundant data. In this case, a large amount of data is certainly not carrying much productive value. The stochastic gradient descent uses only a single example per iteration from the dataset to calculate the gradient descent. The term stochastic indicates that one data example data is picked randomly per iteration.

An artificial neural network is a machine learning computational model. This model is inspired by the biological neural network in the human brain. This model processes the information the same way the human brain processes the information. Artificial neural networks brought some

breakthrough in machine learning research. Some of them are image recognition, computer vision and text processing.

Methodology

The deep learning network depends on multiple parameters that is useful for tuning our network. These parameters help to increase the performance of the model, reduce the memory allocation and training time. These parameters are: 1) batch size, 2) number of epochs, and 3) learning rate.

What is the batch size?

The batch size determines the number of inputs that we feed to the deep learning network for one iteration. The batch size depends on the memory of the machine. When the batch size is too high, it consumes a lot of memory.

H_1: The batch size is inversely proportional to the network accuracy. If we increase the batch size, then accuracy is decreased.

What is an epoch?

The number of epochs determines how many times the whole input set is fed to the neural network. If we increase the number of epochs, then the accuracy of the network will increase.

H_2: The accuracy of a deep learning network is directly proportional to the number of epochs it passes through.

What is a learning rate?

The learning rate determines how frequently the deep learning network changes decision. If the learning rate is too high, then the deep learning network changes decisions very frequently. In most cases, a high learning rate doesn’t bring any drastic changes to the accuracy. Also, a low learning rate may take too long to make a change in decision.

H_3: Lower and higher learning rate do not bring any change to the network. The network performs best within a preferred learning rate.

Dataset Description

Here we used the dataset from Kimia Path24 (Babaie et al., 2017). This dataset has a total of 22,591 training images from 24 different categories. This dataset has a total of 1,325 testing images that include all categories. The image size is 1000 x 1000 for all the images.

Model Description

Data preparation

We used the Keras library from python to build our neural network. First, we loaded the image using the OpenCV library and then resized the image to 28 x 28. Then, we divided the dataset into two parts: 1) training and 2) testing. The training set contained 75% of the data, and the testing set contained 25% of the data. The testing set is used for validating the network. Then, we converted the labels from integer to vectors. After that, we performed the data augmentation. We set the rotation range to 30, width shift range to 0.1, height shift range to 0.1, shear range to 0.2, zoom range to 0.2, horizontal flip to true, and fill mode to nearest.

Building Network

After completing the data augmentation, we built our neural network. Figure 5.4 contains the code snippet of the neural network. Our deep learning model contains 3 convolution networks. A small portion of the convolution layer is used as the input of each node, and the size of the small portion is often 3 x 3 or 5 x 5 (J. Liu et al., 2018). The activation function is the rectified linear unit followed by a flat layer, and finally, two dense layers. The first layer has the activation function rectified linear unit (ReLU) and the output layer has the softmax. The softmax helps to keep the output between zero and one. Then we set the optimizer for this model. We set the adam

optimizer with a learning rate of 10^{-3}. After that, we compiled the model with a batch size of 32 and the number of epochs as 25. We set the loss to binary_crossentropy and metrics to accuracy.

```
class LeNet:
  @staticmethod
  def build(width, height, depth, classes):
    # initialize the model
    model = Sequential()
    inputShape = (height, width, depth)

    # if we are using "channels first", update the input shape
    if K.image_data_format() == "channels_first":
      inputShape = (depth, height, width)

    # first set of CONV => RELU => POOL layers
    model.add(Conv2D(20, (5, 5), padding="same",
      input_shape=inputShape))
    model.add(Activation("relu"))
    model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))

    # second set of CONV => RELU => POOL layers
    model.add(Conv2D(50, (5, 5), padding="same"))
    model.add(Activation("relu"))
    model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))

    # Third set of CONV => RELU => POOL layers
    model.add(Conv2D(64, (5, 5), padding="same"))
    model.add(Activation("relu"))
    model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))

    # first (and only) set of FC => RELU layers
    model.add(Flatten())
    model.add(Dense(64))
    model.add(Activation("relu"))

    # softmax classifier
    model.add(Dense(classes))
    model.add(Activation("softmax"))

    # return the constructed network architecture
    return model
```

Figure 5.4: Building Deep Learning Neural Network with Keras.

Train the model

After setting all the parameters, we ran the model with the fit_generator. Then, we saved the model and the learning curve plot with loss, accuracy, validation loss, and validation accuracy.

Test the model

After we finished training our model, we used a different script to test the model. We loaded the saved model and ran the prediction methods with the test data and generated the confusion matrix to find the accuracy, recall, precision, and f1-score.

Results

The change of accuracy with respect to the batch size is displayed in Table 5.1. From the table, we can see that the accuracy decreases when we increase the batch size, which is satisfying our first hypothesis. Figure 5.5 shows that the decrease in accuracy is linear with respect to the batch size.

Batch Size			
Epoch	**Batch size**	**Learning rate**	**Accuracy**
50	16	0.001	0.6491
50	32	0.001	0.6204
50	64	0.001	0.5585
50	128	0.001	0.4989
50	256	0.001	0.4777
50	512	0.001	0.4438
50	1024	0.001	0.4513

Table 5.1: Change of Accuracy with Batch Size.

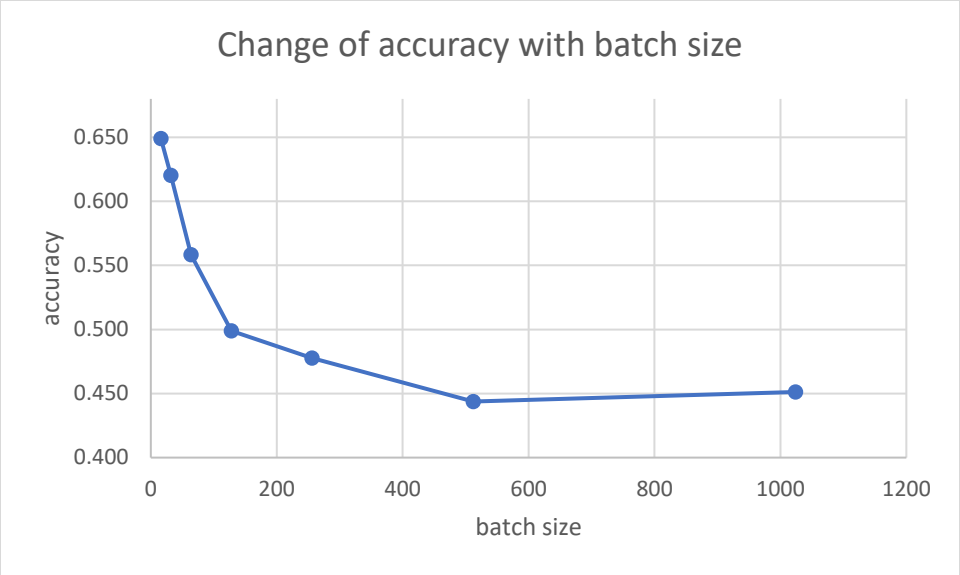

Figure 5.5: Change of Accuracy with Batch Size.

The change of accuracy with the change of epoch is displayed in Table 5.2. From the table, we can see that the accuracy increases with the number of epochs, which satisfies our second hypothesis. Figure 5.6 shows that the accuracy is not linearly correlated with the number of epochs when the value is small. With a larger epoch size, the accuracy increases linearly with respect to the number of epochs.

EPOCH			
Epoch	**Batch size**	**Learning rate**	**Accuracy**
10	32	0.001	0.5668
15	32	0.001	0.5970
25	32	0.001	0.5298
35	32	0.001	0.6128
50	32	0.001	0.6204
70	32	0.001	0.6491
100	32	0.001	0.6732

Table 5.2: Change of Accuracy with Number of Epochs.

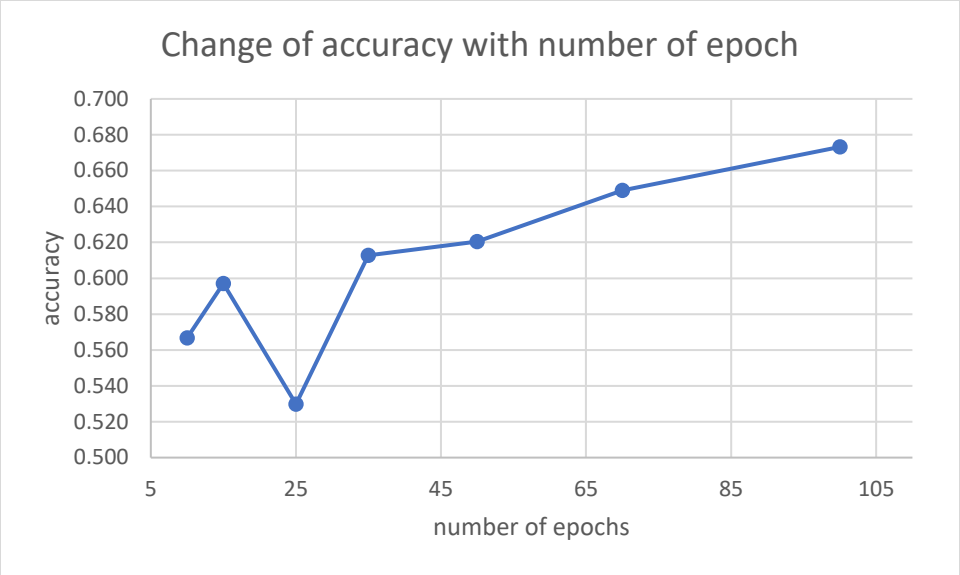

Figure 5.6: Change of accuracy with number of epochs.

From Table 5.3, we can see that the accuracy increases when we increase the learning rate from 0.1 to 0.001. Then after 0.005, the accuracy starts decreasing. At the learning rate of 0.0001, the accuracy is 45.6%, which is less than the accuracy of the learning rate of 0.001 (61.9%). So the accuracy starts decreasing after the learning rate of 0.0001. This experiment validates our third hypothesis that the higher learning rate and lower learning rate do not always bring better changes to the accuracy.

Learning Rate			
Epoch	**Batch size**	**Learning rate**	**Accuracy**
50	32	0.1	0.0528
50	32	0.05	0.0302
50	32	0.01	0.0566
50	32	0.005	0.5925
50	32	0.001	0.6189
50	32	0.0005	0.6272
50	32	0.0001	0.4558

Table 5.3: Change of Accuracy with Respect to Learning Rate.

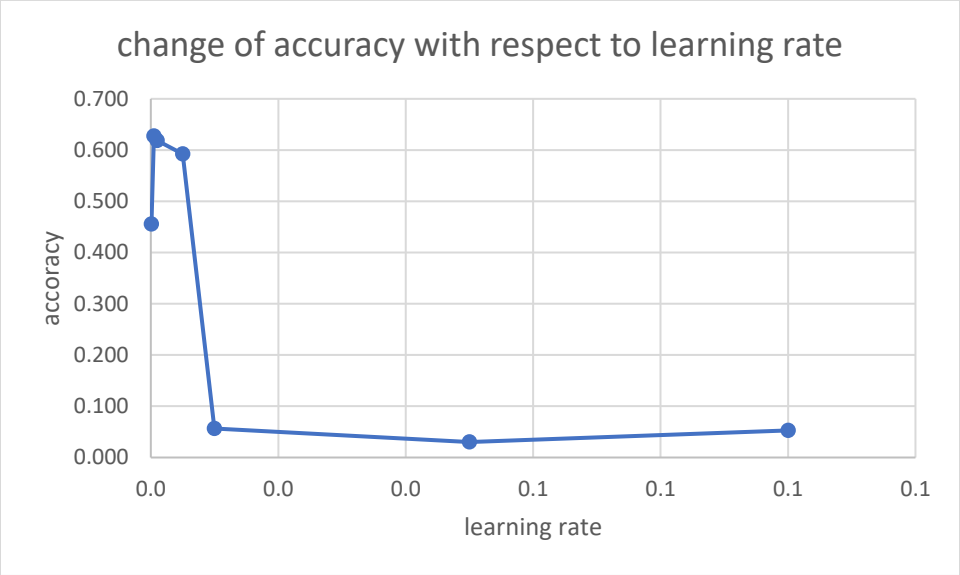

Figure 5.7: Change of Accuracy with Respect to Learning Rate.

Figure 5.7 shows the change in accuracy, with respect to learning rate. The relation between learning rate and accuracy is not linear. The accuracy does not increase or decrease linearly with the change in learning rate.

Discussion

In this experiment, we have done an imperial study of image processing with deep learning. We have conducted the experiments on the KIMIA path24 dataset. We have simply designed the architecture and used the feature selection. The image processing mechanism was used for manipulating the dataset. We have used rotation, width shifting, height shifting shear range, horizontal flip, and fill mode. The network was trained and validated by a total of 22,591 images from the KIMIA path24 dataset. The running time of the network was 6-7 hours depending on the parameters. We used a Google collaboratory training environment equipped with two core Intel's @2.3 GHz Xeon processor with a NVIDIA Tesla K80 (GK210 chipset) (Carneiro et al., 2018).

In this experiment, we used two serial 2-dimensional CNN architectures for segmentation. We used ReLU for the convolution layer and softmax for the fully connected layer. During the compilation, we used the adam optimizer and set loss to binary_crossentropy. The learning rate, the batch size, and the number of epochs was set according to hypotheses testing.

As a limitation of our experiment, various types of network architecture for the segmentation are possible. As proof of the concept to test those hypotheses of our deep learning network parameter, the structure of the architecture was empirically designed after studying the image prototype. An adjustment of our proposed network, including the number of convolution layers, the number of nodes, the activation function, and other parameter upgrades, are possible. As a future work, an optimization of the network architecture and the parameter upgrades could increase the runtime and improve the performance. As our architecture and other parameters were set to test those hypotheses against the KIMIA path24 dataset, this design can be optimized and retrained for other image datasets.

One of the issues of the deep convolutional neural network is sample images. The number of images required to test and validate the network is not always available. The scale of the medical data available for studies is insufficient for machine learning and computer vision. This can affect the performance of deep learning for medical data (Robertson, Azizpour, Smith, & Hartman, 2018). In general, the convolutional neural network requires a large number of training data for each specific category. In our current dataset, we have a total of 24 categories. Some categories have more than 1,000 images for training, whereas few categories have less than 100 images for training and validation. These categories decline in accuracy and affect the overall performance. Using a proper dataset for all categories can improve the performance and provide more degree of freedoms.

Conclusion

In this experiment, we built a simple deep convolutional neural network and Conducted hypothesis testing on three hypotheses on the Kimia Path24 dataset. 2D convolutional neural networks were used for image classification. Our three hypotheses were proved correct. We used an image feature selection and image processing mechanisms for moderating the dataset. The parameter tuning and network structure designs provide essential outcomes, which ensures the credibility of our model.

CHAPTER 6

CONCLUSION

In this work, we have concluded three different studies. The first study is finding an efficient log base entropy for a decision tree dataset with categorical attributes. The second study is finding the most important attributes among all the attributes from a mental health dataset. In the final study, we develop a deep learning model on the Kimia path24 dataset and observe the effect of changing learning parameters, preparing support for all our hypotheses through experimental results. We used 4 different UCI machine learning datasets for our first study. Four datasets were analyzed and compared with different log bases: log_2 , log_e , and log_{10}. Changing the log base has no effect when the dataset has a mixture of continuous and categorical attributes. Our first study suggested that higher values of accuracy were achieved on log_{10} when the dataset contains mostly categorical attributes and those attributes have 3 or more classification elements. In our second study, we developed a detective system that can assist tech companies to help them identify their employees' mental health conditions. Twenty-four different decision trees were used with different parameter values. Among the 24 different decision trees, the model we prefer has an accuracy of 88.5%. The work interference and the family history were counted as the most important attributes for predicting mental health issues. In our final study, we built a simple deep 2D convolutional neural network for image classification. We developed three different hypotheses and tested those hypotheses on the Kimia path24 dataset. All hypotheses made on our final study were supported successfully. The credibility of our model was tested by varying parameter tuning and design structure. The discrepancy of the dataset (not having the same number of images for all the categories) affects the overall performance of our model and reduces the

inclination of accuracy. This can be avoided by using the same amount of data for all the categories.

REFERENCES

Affonso, C., Rossi, A. L. D., Vieira, F. H. A., & de Leon Ferreira, A. C. P. J. E. S. w. A. (2017). Deep learning for biological image classification. *85*, 114-122.

Babaie, M., Kalra, S., Sriram, A., Mitcheltree, C., Zhu, S., Khatami, A., . . . Tizhoosh, H. R. (2017). *Classification and retrieval of digital pathology scans: A new dataset.* Paper presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops.

Bao, S., & Chung, A. C. (2018). Multi-scale structured CNN with label consistency for brain MR image segmentation. *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization, 6*(1), 113-117.

BLAS, N. (2014). Retrieved from http://www.netlib.org/blas/

Brosch, T., & Tam, R. (2013). *Manifold learning of brain MRIs by deep learning.* Paper presented at the International Conference on Medical Image Computing and Computer-Assisted Intervention.

Brosch, T., Yoo, Y., Li, D. K., Traboulsee, A., & Tam, R. (2014). *Modeling the variability in brain morphology and lesion distribution in multiple sclerosis by deep learning.* Paper presented at the International Conference on Medical Image Computing and Computer-Assisted Intervention.

Carneiro, T., Da Nóbrega, R. V. M., Nepomuceno, T., Bian, G.-B., De Albuquerque, V. H. C., & Reboucas Filho, P. P. (2018). Performance analysis of google colaboratory as a tool for accelerating deep learning applications. *IEEE Access, 6*, 61677-61685.

Chen, H., Dou, Q., Yu, L., & Heng, P.-A. (2016). Voxresnet: Deep voxelwise residual networks for volumetric brain segmentation. *arXiv preprint arXiv:1608.05895*.

Chen, Y., Lin, Z., Zhao, X., Wang, G., & Gu, Y. (2014). Deep learning-based classification of hyperspectral data. *IEEE Journal of Selected topics in applied earth observations and remote sensing, 7*(6), 2094-2107.

Chetlur, S., Woolley, C., Vandermersch, P., Cohen, J., Tran, J., Catanzaro, B., & Shelhamer, E. (2014). cudnn: Efficient primitives for deep learning. *arXiv preprint arXiv:1410.0759*.

Choi, H., & Jin, K. H. (2016). Fast and robust segmentation of the striatum using deep convolutional neural networks. *Journal of neuroscience methods, 274*, 146-153.

Cireşan, D. C., Giusti, A., Gambardella, L. M., & Schmidhuber, J. (2013). *Mitosis detection in breast cancer histology images with deep neural networks.* Paper presented at the International conference on medical image computing and computer-assisted intervention.

de Brebisson, A., & Montana, G. (2015). *Deep neural networks for anatomical brain segmentation.* Paper presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops.

Deng, L. (2014). A tutorial survey of architectures, algorithms, and applications for deep learning. *APSIPA Transactions on Signal and Information Processing, 3*.

Dong, C., Loy, C. C., He, K., & Tang, X. (2014). *Learning a deep convolutional network for image super-resolution.* Paper presented at the European conference on computer vision.

Ghaznavi, F., Evans, A., Madabhushi, A., & Feldman, M. (2013). Digital imaging in pathology: whole-slide imaging and beyond. *Annual Review of Pathology: Mechanisms of Disease, 8*, 331-359.

Gnanlet, A., & Gilland, W. G. (2009). Sequential and simultaneous decision making for optimizing health care resource flexibilities. *Decision Sciences, 40*(2), 295-326.

Gupta, A., Wilkerson, G. B., Sharda, R., & Colston, M. A. (2018). Who is More Injury-Prone? Prediction and Assessment of Injury Risk. *Decision Sciences*.

Gupta, S., Agrawal, A., Gopalakrishnan, K., & Narayanan, P. (2015). *Deep learning with limited numerical precision.* Paper presented at the International Conference on Machine Learning.

Havaei, M., Davy, A., Warde-Farley, D., Biard, A., Courville, A., Bengio, Y., . . . Larochelle, H. (2017). Brain tumor segmentation with deep neural networks. *Medical image analysis, 35*, 18-31.

Ho, J., Parwani, A. V., Jukic, D. M., Yagi, Y., Anthony, L., & Gilbertson, J. R. (2006). Use of whole slide imaging in surgical pathology quality assurance: design and pilot validation studies. *Human pathology, 37*(3), 322-331.

Karpathy, A. (2019). CS231n: Convolutional Neural Networks for Visual Recognition. Retrieved from http://cs231n.github.io/neural-networks-1/#actfun

Kim, J., Calhoun, V. D., Shim, E., & Lee, J.-H. (2016). Deep neural network with weight sparsity control and pre-training extracts hierarchical features and enhances classification performance: Evidence from whole-brain resting-state functional connectivity patterns of schizophrenia. *Neuroimage, 124*, 127-146.

Kuhn, M., & Johnson, K. (2013). *Applied predictive modeling* (Vol. 26): Springer.

Larose, D. T., & Larose, C. D. (2015). *Data mining and predictive analytics*: John Wiley & Sons.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature, 521*(7553), 436.

Litjens, G., Kooi, T., Bejnordi, B. E., Setio, A. A. A., Ciompi, F., Ghafoorian, M., . . . Sánchez, C. I. J. M. i. a. (2017). A survey on deep learning in medical image analysis. *42*, 60-88.

Litjens, G., Sánchez, C. I., Timofeeva, N., Hermsen, M., Nagtegaal, I., Kovacs, I., . . . Van Der Laak, J. (2016). Deep learning as a tool for increased accuracy and efficiency of histopathological diagnosis. *Scientific reports, 6*, 26286.

Liu, J., Pan, Y., Li, M., Chen, Z., Tang, L., Lu, C., . . . Analytics. (2018). Applications of deep learning to MRI images: A survey. *1*(1), 1-18.

Liu, S., Liu, S., Cai, W., Pujol, S., Kikinis, R., & Feng, D. (2014). *Early diagnosis of Alzheimer's disease with deep learning.* Paper presented at the 2014 IEEE 11th international symposium on biomedical imaging (ISBI).

Liu, W., Chawla, S., Cieslak, D. A., & Chawla, N. V. (2010). *A Robust Decision Tree Algorithm for Imbalanced Data Sets.* Paper presented at the SIAM International Conference on Data Mining.

Ludwig, S. A., Jakobovic, D., & Picek, S. (2015). *Analyzing gene expression data: Fuzzy decision tree algorithm applied to the classification of cancer data.* Paper presented at the Fuzzy Systems (FUZZ-IEEE), 2015 IEEE International Conference on.

Meyer, G., Adomavicius, G., Johnson, P. E., Elidrisi, M., Rush, W. A., Sperl-Hillen, J. M., & O'Connor, P. J. (2014). A machine learning approach to improving dynamic decision making. *Information Systems Research, 25*(2), 239-263.

Ngiam, J., Khosla, A., Kim, M., Nam, J., Lee, H., & Ng, A. Y. (2011). *Multimodal deep learning.* Paper presented at the Proceedings of the 28th international conference on machine learning (ICML-11).

Obenshain, M. K. (2004). Application of data mining techniques to healthcare data. *Infection Control & Hospital Epidemiology, 25*(8), 690-695.

Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z. B., & Swami, A. (2016). *The limitations of deep learning in adversarial settings.* Paper presented at the 2016 IEEE European Symposium on Security and Privacy (EuroS&P).

Payan, A., & Montana, G. (2015). Predicting Alzheimer's disease: a neuroimaging study with 3D convolutional neural networks. *arXiv preprint arXiv:1502.02506*.

Payan, A., & Montana, G. J. a. p. a. (2015). Predicting Alzheimer's disease: a neuroimaging study with 3D convolutional neural networks.

Pinaya, W. H., Gadelha, A., Doyle, O. M., Noto, C., Zugman, A., Cordeiro, Q., . . . Sato, J. R. (2016). Using deep belief network modelling to characterize differences in brain morphometry in schizophrenia. *Scientific reports, 6*, 38897.

Plis, S. M., Hjelm, D. R., Salakhutdinov, R., Allen, E. A., Bockholt, H. J., Long, J. D., . . . Calhoun, V. D. J. F. i. n. (2014). Deep learning for neuroimaging: a validation study. *8*, 229.

Qin, B., Xia, Y., Prabhakar, S., & Tu, Y. (2009). *A rule-based classification algorithm for uncertain data.* Paper presented at the Data Engineering, 2009. ICDE'09. IEEE 25th International Conference on.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*: Elsevier Science.

Razzak, M. I., Naz, S., & Zaib, A. (2018). Deep learning for medical image processing: Overview, challenges and the future. In *Classification in BioApps* (pp. 323-350): Springer.

Robertson, S., Azizpour, H., Smith, K., & Hartman, J. J. T. R. (2018). Digital image analysis in breast pathology—from image processing techniques to artificial intelligence. *194*, 19-35.

Rutkowski, L., Jaworski, M., Pietruczuk, L., & Duda, P. (2014). The CART decision tree for mining data streams. *Information Sciences*, 1–15.

Salzberg, S. L. (1994). C4. 5: Programs for machine learning by j. ross quinlan. morgan kaufmann publishers, inc., 1993. In: Kluwer Academic Publishers.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural networks, 61*, 85-117.

Services, M. S. S. A. (2018). Microsoft Decision Trees Algorithm Technical Reference. Retrieved from https://docs.microsoft.com/en-us/analysis-services/data-mining/microsoft-decision-trees-algorithm-technical-reference?view=asallproducts-allversions

Suk, H.-I., Lee, S.-W., Shen, D., & Initiative, A. s. D. N. (2014). Hierarchical feature representation and multimodal fusion with deep learning for AD/MCI diagnosis. *Neuroimage, 101*, 569-582.

Suk, H.-I., Wee, C.-Y., Lee, S.-W., & Shen, D. (2016). State-space model with deep learning for functional dynamics estimation in resting-state fMRI. *Neuroimage, 129*, 292-307.

Wang, Z., Chen, J., Hoi, S. C. J. I. T. o. P. A., & Intelligence, M. (2020). Deep learning for image super-resolution: A survey.

Yana, R., Maa, Z., Zhaob, Y., & Kokogiannakis, G. (2016). A decision tree based data-driven diagnostic strategy for air handling units. *Energy and Buildings*, 37-45.