A Dissertation

entitled

Development of Adaptive Computational Algorithms for Manned and Unmanned Flight Safety

by

Colin P. Elkin

Submitted to the Graduate Faculty as partial fulfillment of the requirements for the Doctor of Philosophy Degree in Engineering

---

Dr. Vijay Devabhaktuni, Committee Chair

Dr. Mansoor Alam, Committee Member

Dr. Ahmad Javaid, Committee Member

Dr. Devinder Kaur, Committee Member

Dr. Weiqing Sun, Committee Member

Dr. Lawrence Thomas, Committee Member

Dr. Cyndee Gruden, Dean
College of Graduate Studies

The University of Toledo
December 2018

An Abstract of

Development of Adaptive Computational Algorithms for Manned and Unmanned Flight Safety

by

Colin P. Elkin

Submitted to the Graduate Faculty as partial fulfillment of the requirements for the Doctor of Philosophy Degree in Engineering

The University of Toledo
December 2018

A strong emphasis on safety in commercial and military aviation is as old and as significant as the field of aviation itself. With the growing role of autonomy in aviation, the future of flight comprises of two general directions: manned and unmanned. Manned aircraft is the more established area, in which a human flight crew serves as the main driving force in ensuring an aircraft's safety and success. Within this time-tested concept, the most significant bottleneck of safety lies within a crew managing tasks of high mental workload. In recent years, autonomy has aided in easing cognitive workload. From there, the challenge lies within applying a seamless blend of human and autonomous control based on the needs of one's mental load. Meanwhile, the field of unmanned aerial vehicles (UAVs) poses its own unique challenges of integrating into a shared airspace and transitioning from remote human-centric control to fully autonomous control. In such a case, minimizing discrepancies between predicted UAV behavior and actual outcomes is an ongoing task to ensure a safe and reliable flight.

While manned and unmanned flight safety may seem distinctly different in these regards, this dissertation proposes an overarching common theme that lies within the ability to effectively model inputs and outputs through machine learning to predict potential safety hazards and thereby improve the overall flight experience. This

process is conducted by 1) evaluating different machine learning techniques on assessing cognitive workload, 2) predicting trajectories for autonomous UAVs, and 3) developing adaptive systems that dynamically select appropriate algorithms to ensure optimal prediction accuracy at any given time.

The first phase of the research involves the manned side of flight safety and does so by examining effects of different machine learning techniques used for assessing cognitive workload. This begins by comparing the different algorithms on four different datasets involving cognitive activity based on physiological and subjective data. From there, two new algorithms are developed that dynamically select a machine learning technique based on the attributes of the given physiological data: one that statically chooses a method and another that dynamically changes methods over time based on which is projected to provide optimal accuracy and efficiency. By being able to accurately classify an activity with a certain amount of expected cognitive load, this can be applied in aircraft to assist pilots in early detection of mental overload and underload.

The second phase then invokes unmanned flight safety by aiming to enhance prediction of autonomous UAV data. This is done by fusing navigational coordinates with radar data (e.g. how close a UAV is to another vehicle or an obstacle) using Dempster-Shafer Evidence Theory. The algorithm is then compared against other mechanisms for UAV data prediction with encouraging results. Finally, an additional algorithm is developed that dynamically chooses methods at different points in time based on which is expected to produce the best accuracy. Thus, by improving accurate predictions of future UAV data, unmanned flight safety can be enhanced by minimizing discrepancies between expectations and outcomes in an increasingly unpredictable shared airspace.

To my fiancé, Mary, for her past, present, and future love and support.

Acknowledgments

The quantity and extent of my gratitude to all of those who made my graduate studies possible could easily be a dissertation of its own. Regardless, I must first of all commend my advisor, Dr. Vijay Devabhaktuni, for consistently going above and beyond in his guidance, leadership, support, motivation, friendship, and patience over the past four-and-a-half years.

Secondly, I must acknowledge the Air Force Research Laboratory (AFRL), which funded my dissertation research in the forms of a Dayton Area Graduate Studies Institute (DAGSI) fellowship and an additional grant subcontracted through Soar Technology, Inc. Thus, I also thank my DAGSI sponsor Dr. Gregory Funke as well as Mr. Jack Zaientz at Soar Tech for their periodic guidance. Other sources to be acknowledged are the National Science Foundation and the University of Toledo Department of Engineering Technology for funding the first year of my doctoral program.

I also express gratitude towards Drs. Mansoor Alam, Ahmad Javaid, Devinder Kaur, Weiqing Sun, and Lawrence Thomas for serving on my dissertation committee In addition, I wish to thank Dr. Kevin Xu for his periodic assistance as well as for providing access to two vital datasets. I also want to thank my former lab mates of NE 1024, 2033, and 2042.

On a personal note, I must express gratitude to my parents John and Rebecca Elkin and to my in-laws Sam and Rebecca Lucio for all they have done near and far. Finally, I must above all thank my fiancé Mary all for her love, care, and patience over the past four years.

Contents

List of Tables

List of Figures

List of Abbreviations

2D	two-dimensional
3D	three-dimensional
ACO	ant colony optimization
ANN	artificial neural network
BPA	basic probability assignment
CDF	cumulative distribution function
CPU	central processing unit
CSV	comma separated value
DEAP	A Database for Emotion Analysis using Physiological Signals
DNN	deep neural network
DSET	Dempster-Shafer Evidence Theory
DT	decision tree
ECG	electrocardiography
EDA	electrodermal activity
EEG	electroencephalogram
EKF	extended Kalman filter
fNIR	functional near-infared spectroscopy
GB	gigabyte(s)
GHz	gigahertz
GNU	GNU is not UNIX
GPL	general public license
GPS	global positioning system
INS	inertial navigation system
IPP	Imprecise Propagation Probability
KNN	k-nearest neighbor
L-BFGS	Limited-memory Broyden–Fletcher–Goldfarb–Shanno algorithm
LiDAR	light detection and ranging
MATLAB	MATrix LABoratory
ML	machine learning
MLP	multi-layer perceptron

MSE	mean squared error
PDF	portable document format
RAM	random access memory
RBF	radial-basis function
RF	random forest
RMSE	root mean squared error
SAPHYRE	Sliding-Scale Autonomy through Physiological Rhythm Evaluations
SGD	stochastic gradient descent
SVM	support vector machine(s)

List of Symbols

$\mathbf{x}$ ..... vector x
$P(x)$ ..... probability of x
$f(x)$ ..... function of x
$Y|X$ ..... outcome Y given X
Θ ..... frame of discernment
$\oplus$ ..... exclusive or
Bel ..... belief
Pl ..... plausibility
$\subset$ ..... subset
$\cap$ ..... intersection
$\emptyset$ ..... empty set

Chapter 1

Introduction

Due to the ongoing evolution of artificial intelligence and autonomy, the field of aviation is undergoing rapid changes. More specifically, aviation appears to be branching into two primary subfields. One is the time-tested concept of manned aircraft, which involves a human pilot in many commercial and military applications, although this area is being improved by the addition of autonomy to aid the flight crew in the successful execution and completion of a flight. The other is the more recent development of unmanned aircraft, which involves smaller unmanned aerial vehicles (UAVs) that are piloted remotely.

While early UAVs have been almost entirely reliant on remote human control, the concept of autonomy has grown rapidly in recent years regarding its role in UAV piloting, even to the point of fully autonomous UAVs. Figure 1 depicts a spectrum of flight categories, ranging from fully human-driven to fully autonomous. This includes some intermediate categories such as sliding scale autonomy [1] to aid human pilots with a select amount of automated control or remotely piloted UAVs that involve both remote human control and some levels of autonomy. The overall scope of this research, however, focuses primarily on the two edges of the spectrum.

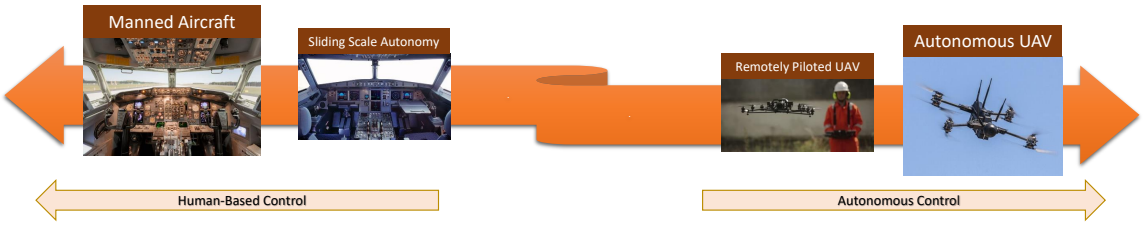


Figure 1-1: Evolution of aircraft control from human to autonomy.

1.1 Flight Performance and Safety

Over the entire lifespan on manned aircraft, safety culture [2–4] has been a critical and overarching necessity throughout every facet of the design and planning of commercial and noncommercial flights, often serving as an important factor in customers' perception of flight safety [5, 6]. Despite this priority, a significant challenge lies within the ability of a crew member to effectively manage a high level of mental workload. This is primarily prevalent under pilots and air traffic controllers who work with enormous amounts of stress and multitasking. This results in a high amount of cognitive workload, which adversely affects an individual's situational awareness as well as one's ability to perform all given tasks effectively. In recent years, increases in partial autonomy have been introduced to ease a pilot's workload [7].

The underlying challenge, however, lies within seamlessly transitioning between human control and autonomous control for a particular task in order to avoid counterproductivity caused by excess complexity and lack of understanding of the automation [8], which could in turn lead to increased cognitive load from the autonomy itself [9]. For this, it is beneficial to detect and predict an individual's cognitive load in order to achieve early detection of cognitive overload (e.g. a pilot is overwhelmed with too many tasks requiring significant amounts of precision) or underload (e.g. so little is happening that a pilot's senses are dulled and he or she cannot respond as quickly to sudden changes of events).

In predicting cognitive workload, many research efforts have sought to model cognitive workload using physiological and objective input data. These include many established machine learning techniques, such as artificial neural networks, support vector machines, and decision trees. By training any of these algorithms with appropriate data, early detection of cognitive workload levels can be utilized to seamlessly assist a human pilot or other crew member with intense levels of airline-related stress and multitasking.

1.1.1 From Human Pilots to Autonomous Pilots

As for unmanned aircraft, the challenges of human operators and cognitive workload are still prevalent, particularly when coordinating multiple UAVs simulatneously [10]. However, a more recent trend in autonomous UAVs indicates a need to predict partly [11] or even fully [12] autonomous behavior. By working to minimize the discrepancies between predicted behavior (e.g. trajectories, actions, response to obstacles or other UAVs), these vehicles can benefit from improved safety, particularly due to the growing need for a standardized shared airspace [12–15] that places UAVs into a similar frame of safety as those of manned aircraft.

As in the case of manned aircraft, accurate modeling of input and output data can assist in effectively predicting future UAV behavior. By enhancing the accuracy of these predictions, pilots and mission planners can have a clearer idea of what is needed for a UAV to safely and successfully complete any mission objectives. In such cases, however, machine learning alone is not always enough to accurately model complex behavior, especially when many different types of data are involved. Hence, the concept of data fusion becomes one of great importance. Thus, by strategically applying both machine learning and data fusion, autonomous UAV behavior can be modeled in a way that are there fewer surprises and deviations between expectations and outcomes, thereby maximizing unmanned flight safety in increasingly complex

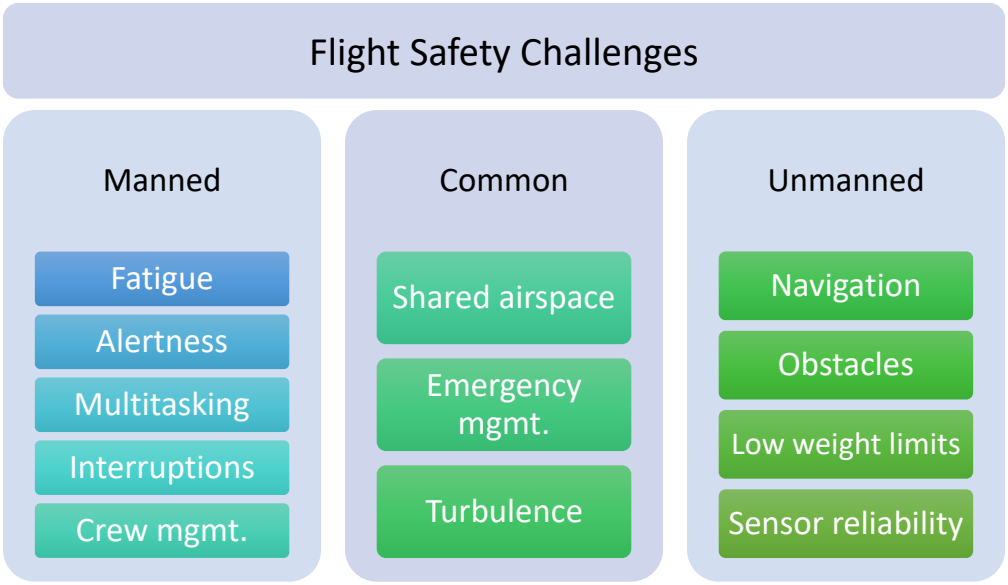


Figure 1-2: Comparison of flight safety challenges for manned and unmanned aircraft.

mission scenarios.

1.2 Motivations

Placing focus on both manned and unmanned flight safety helps to identify and address the similarities that exist between the two types of aviation. Figure 2 provides a brief comparison of the challenges involved in both areas. As their differences indicate, the challenges of manned aircraft are based primarily on human factors, such as fatigue, alertness, and crew management. Most of the unmanned challenges, however, are based on the UAV itself, including low weight limits, obstacles, and reliability of sensors [16]. The abundance of shared challenges, however, including the issues of a shared airspace and emergency management, serve as the unified core of this research.

Most importantly, there are to date no current research publications that address manned and unmanned flight safety from a common lens. This observation is surpris-

ing given the above challenges and the technical milestones that have been applied to each from the perspective of machine learning. That is, given some kind of flight data, whether physiological from a pilot or sensory from an autonomous entity, this form of data science has played a crucial role in making predictions, classifications, and analysis on the state of either form of flight safety.

Thus, given such a background, this research aims to confirm that the use of machine learning can effectively aid in both focus areas in the predictions of relevant outcomes (e.g. decreasing a pilot's cognitive workload, altering the intended trajectory of a UAV), which in turn can be uniquely assessed to improve flight safety.

1.3 Research Objectives

To thoroughly and effectively address both sides of flight safety improvement, this research proposes the following process:

- Identify different machine learning techniques used in assessing cognitive workload and test them on multiple physiological datasets to properly correlate physiological and objective inputs with different mental activities of varying cognitive load
- Perform an analysis of alternatives on the different techniques based on the variety of results to determine the pros and cons of one algorithm over another under different scenarios
- Develop a technique that dynamically chooses a machine learning algorithm based on the attributes of a given dataset and can change such an algorithm at different points of time if needed
- Apply this novel technique in a way that it can alert pilots of potential future overload or underload, at which point steps can be taken to prevent such an

imbalance, thereby improving human-side flight safety

- Examine the effects of different machine learning techniques on autonomous UAV data and incorporate data fusion to combine multiple types of dissimilar data and improve prediction accuracy
- Develop a technique that dynamically selects a prediction algorithm based on potential prediction accuracy and can switch methods over time when potential accuracy crossovers occur
- Apply this technique on UAV data to ensure smoother autonomous path planning and behavior, thereby improving unmanned flight safety

By achieving these goals, greater improvements in aviation safety can be achieved in order to overcome many of the shared challenges on the manned and unmanned sides as well as their own unique challenges.

1.4 Publications and Contributions to Dissertation

The major publications and contributions to this research are presented in Table 1.1.

1.5 Dissertation Organization

This dissertation unfolds as follows:

Chapter 2 comprises the literature review of the dissertation. It provides a more in-depth look at past and present practices and challenges associated with flight safety. Also included are established uses of machine learning in cognitive workload assessment as well as their applications, the growing role of autonomy in flight, and established computational approaches for UAV data prediction.

Chapter 3 provides a theoretical overview of machine learning techniques relevant to cognitive workload assessment, a description and analysis of the relevant physiological datasets used in this end of the research, a thorough look at the results under numerous dimensions of variability, and a cost-benefit analysis of the different techniques under different conditions.

Chapter 4 describes the premise and goals for implementing the previous techniques into an adaptive system. Also included are an algorithmic overview, results, discussion, and the resulting implications for the future of manned flight safety.

Chapter 5 contains the motivations for seeking enhanced UAV data prediction, an overview of the data fusion process, a description of the established competing techniques that are to be utilized for comparison, thorough results under the many dimensions of variability, and meaningful discussion.

Chapter 6 details the premise and goals for implementing the previous techniques into an adaptive system (similarly to Chapter 4 but for unmanned flight), an algorithmic overview, results under most of the same dimensions of variability, discussion, and implications for the future of unmanned flight safety.

Chapter 7 draws final conclusive remarks and discusses potential future ambitions in which this research could take shape.

Finally, this dissertation ends with an appendix, providing instructions on how to obtain any or all of the project source code.

Table 1.1: Publications and contributions to dissertation.

Type	Publication/Contribution
Journal Paper	Computational approaches for optimal manned and unmanned flight safety: a survey.
Conference Paper	Fundamental cognitive workload assessment: a machine learning comparative approach.
Conference Paper	Analysis of alternatives for neural network training techniques in assessing cognitive workload.
Journal Paper	Analysis of alternatives for machine learning in assessing cognitive workload.
Conference Paper	Adaptive machine learning approach for assessing cognitive workload.
Journal Paper	Enhancing human-driven flight safety through early detection of cognitive overload.
Journal Paper	Prediction of unmanned aerial vehicle behavior through a combination of Dempster-Shafer and artificial neural network sensor fusion.
Conference Paper	Adaptive machine learning system for enhanced UAV path prediction.
Journal Paper	Improving unmanned flight safety through adaptive machine learning of trajectory estimation.
Source Code	A re-usable Python library containing the source code of the algorithmic implementations for the manned phase of the research.
Source Code	A re-usable MATLAB library containing the source code of the algorithmic implementations for the unmanned phase of the research.

Chapter 2

Computational Approaches for Optimal Manned and Unmanned Flight Safety

This chapter consists of a comprehensive literature review. This includes general flight safety practices and challenges, the evolution from human control to autonomy, an overview of cognitive workload in manned and unmanned aircraft, established roles of machine learning in workload assessment, autonomy in flight, and methods for assessing autonomy. A key observation is that in examining prior research on flight safety, a wide variety of prior works exist on both the manned and unmanned sides, but none appear to examine both areas together. The findings and analyses of the literature in these areas serve as the bases for the many technical phases of this dissertation.

2.1 Flight Safety Practices and Challenges

The concept of flight safety itself is a topic that has been prominent in literature for decades that has been approached from a variety of angles. For instance, Camp-

bell and Lahey wrote a survey in 1984 that details aircraft accidents that resulted from fatigue failures in aircraft [17]. More specifically, these accidents are the result of hardware failure in various parts of the aircraft that can be resolved by design, testing, load monitoring, inspection, and part replacements in that order. In 1990, a publication by Wycoff and Holley examined the effects of pasengers' perception of an airline and its flight safety based on being touched on the shoulder or forearm by a flight attendant [18]. In a more general context, O'Hare and Chalmers published a 1999 study comparing accident rates from multiple parameters, including categories of aircraft and hazardous conditions such as low fuel [19]. Other works include a 2001 link between safety attitudes with flight performance [20], the proposal of a new safety index in 2004 based on relative safety level from the perspective of risk and competitive safety [21], and improvement of flight safety based on prediction of bird migration patterns [22] in 2007. A 2009 work by Macrae oversaw a need for improved risk management by increasing the visbility of potential threats [23]. By 2011, thorough reviews of measuring aviation safety climate from a wide range of variables are more prominent [24].

Overall, a key limitation in human-driven flight is the general concept of human error [25–27]. One such cause of error is pilot fatigue [28–30]. This can be caused by a variety of factors, such as regional airline operations (e.g. regulatory differences, scheduling practices) [31], long haul flight operations (e.g. rapid time zone changes, sleep disturbances) [32], and corporate flight operations (e.g. unscheduled flights, extended duty days) [33]. Other key drivers are the attitudes and safety cultures of a flight crew [4, 20] as well as weather-related issues [34]. Overall, these limitations can lead to reduced task engagement [35], slower movement, increased mistakes, and memory difficulties [29] as well as general confusion and cognitive impairment [36]. Another key deficiency is the effect of concurrent task demands [37], or the need to run multiple tasks at the same time that are frequently interrupted.

It should also be noted that pilots are not the only members of the flight crew that endure large amounts of stress. Other humans affected include air traffic controllers [38–43] and even flight attendants [44].

2.2 Optimizing Human Control through Cognitive Workload Assessment

The ability to effectively manage a pilot's mental workload under periods of stress [45–47], fatigue, and high levels of multitasking is key to ensuring successful human performance. Thus, the ability to make classifications, predictions, and assessments of cognitive workload in essential applications remains an ongoing research challenge. More recently, a variety of machine learning algorithms have become available that address cognitive workload assessment quite well. These range from support vector machines and decision trees to k-nearest neighbors and artificial neural networks.

2.2.1 Cognitive Workload

Cognitive workload refers to the overall amount of effort being utilized in working memory. In a certain light, the concept can be regarded as a type of balancing act between underload and overload of such memory. For instance, many prior human factors research efforts have sought to address the problem of overload [48], such as when an individual is given tasks that are too numerous or too difficult, making one's overall performance of such work less than optimal.

Based on prior research efforts, detection and prevention of mental overload appear to be the more established objectives, particularly in applications such as general cognitive tests [48], piloting of remotely operated vehicles [49], science education [50], team-based command and control environments [51], and even ordinary driving

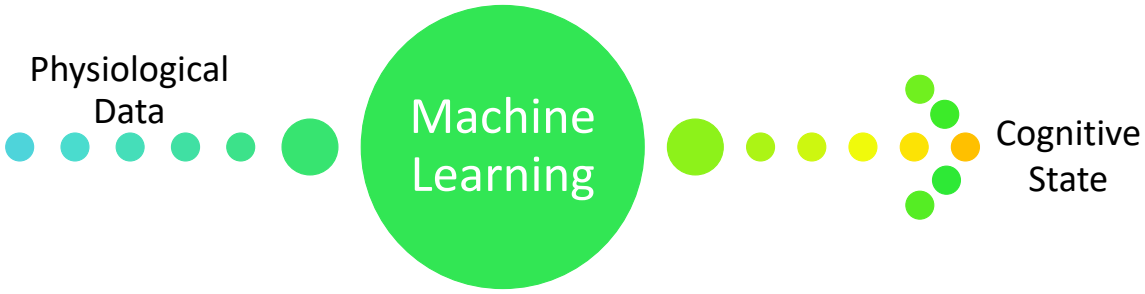


Figure 2-1: Role of machine learning in the assessment of cognitive workload.

[52–54]. Overload generally occurs when one must undergo numerous parallel tasks and/or difficult tasks, which results in increased human error.

Conversely, however, the possibility of underload [55] must also be addressed. For instance, a traveler engaging in a long and monotonous drive may in fact benefit from increasing mental workload, as a low amount may find the driver in a "highway hypnosis," dulling his senses and leaving him unable to react effectively to sudden changes, such as the immediate slowdown of a leading car.

Overall, the ability to accurately and effectively assess one's cognitive state has remained a vital challenge in the field of neuroergonomics. Furthermore, the desire to predict, classify, and otherwise manipulate cognitive workload assessment has been greatly aided by the many novel facets of machine learning.

2.2.2 Machine Learning

Machine learning is a broad and well-recognized research field that spans across a wide variety of applications. Cognitive workload is no exception and has been

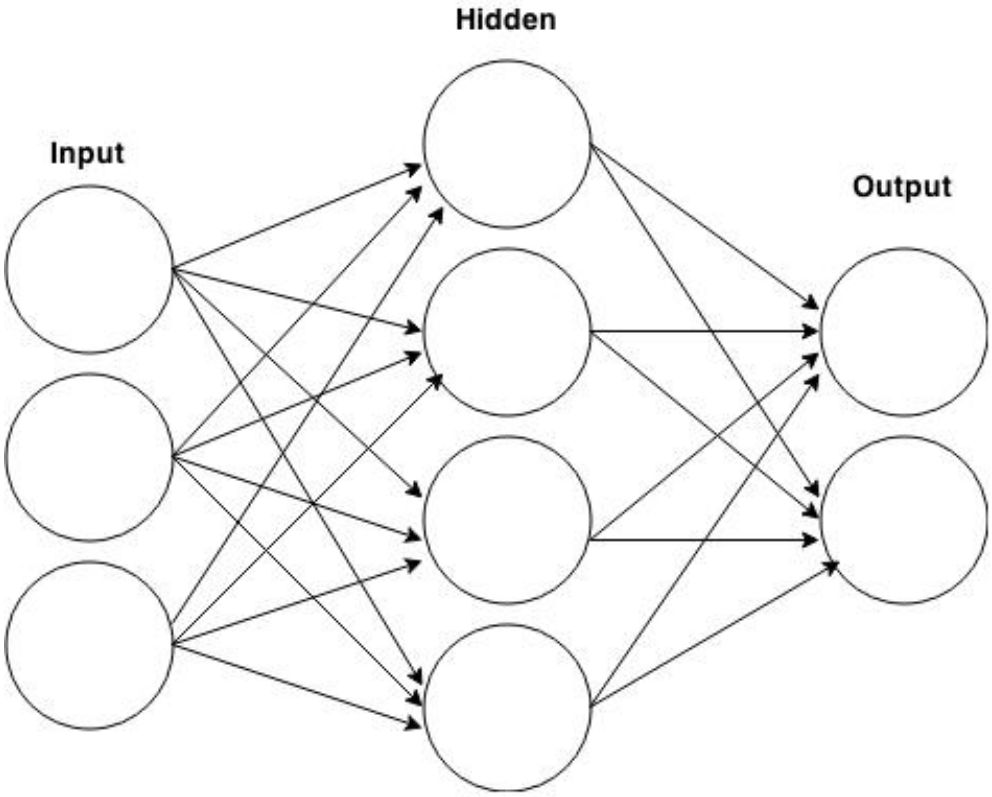


Figure 2-2: Overview of an artificial neural network.

addressed by this field in a vast plethora of ways.

Like with any machine learning problem, the key to forming an accurate and effective model is to properly decide the relevant inputs and outputs. Inputs in this field are typically physiological in nature and have included heart rate [56, 57], respiratory [58], electrocardiographic (ECG) [58, 59], eye movement [10], near-infrared spectroscopy (fNIR) [60, 61], magnetoencephalographic [62], and electroencephalogram (EEG) [48, 63–65] data, the latter of which seems to be the most common [58, 65, 66]. Electrodermal activity (EDA) also appears to be a promising physiological input in cognitive activity [67]. In the context of pilot workload, heart rate, heart rate variability, blood pressure, and respiration are common physiological metrics [68–70].

2.2.2.1 Artificial Neural Networks

An artificial neural network (ANN) is a common machine learning algorithm based on biological neural networks. Many types of supervised ANNs, such as multilayer perceptron (MLP), consist of multiple layers of nodes. In the case of MLP, these neural networks consist of input, output, and hidden nodes [71].

The multilayer perceptron (MLP) feed forward equations are [72]

$$y_k = v_{0k} + \sum_{j=1}^{n_h} v_{jk} z_j \tag{2.1}$$

and

$$\gamma_j = u_{0j} + \sum_{i=0}^{n_i} u_{ij} x_i, \tag{2.2}$$

where $k = 1, 2, \ldots, n_o$ in Equation 2.1 and $j = 1, 2, \ldots, n_h$ in Equation 2.2. n_i, n_h, and n_o are the numbers of input, hidden, and output neurons, respectively. Furthermore, all x_i values are based on given input values in a particular sample of data, and z_j is based on the sigmoid activation function

$$z_j = \frac{1}{1 + e^{-\gamma_j}}. \tag{2.3}$$

Once the neural network architecture, i.e. set of weights and biases, is decided, the weights are initialized while the model output is calculated. These variables can be trained using supervised techniques such as quasi-Newton, stochastic gradient descent, or the limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) algorithm.

ANNs appear to be most established in a variety of cognitive workload problems, including classification of mental workload [58, 73, 74], qualitative analysis based on air traffic data [41], cross-task workload [58, 75], and classification of crewmember workload [76]. A subset of ANNs known as deep neural networks (DNNs) has been

applied to similar but differing applications, such as emotion recognition [66], brain-computer interfaces [77], and analysis of human activity [78].

In supervised ANNs, two of the most established optimization methods include quasi-Newton and stochastic gradient descent. Regarding cognitive load, the latter training technique has modeled temporal dependencies using memory structures [79] and workload model performance [43], while the former aided in inferring mental load based on pupillary dilation [80] and both methods served as classifiers in decision support sys-tems [81]. It is worth noting that only some of these applications have used these optimizers specifically for ANNs while others have utilized these as standalone numerical methods. Other training methods include global techniques such as ant colony optimization [72], in which a biologically inspired technique is based on the behavior of ants. For unsupervised ANNs, restricted Boltzmann machines have found success in assessing cognitive workload across multiple tasks [82]. Relative to assessing pilot workload, ANNs have been used in evaluation within a flight simulator based on ECG measurements [83].

2.2.2.2 Support Vector Machines

Support vector machines (SVMs) are a two-group classification problem that maps data points onto a high dimensional space [84]. In this technique, a support vector network maps multiple input factors into such a space through nonlinear mapping techniques [84]. The term support vector refers to the vectors that determine the largest margin of separation between two groups. In general, SVMs offer fast training in a distributed manner with efficient use of processing resources. Due to the relatedly binary nature of SVM outputs, this technique is more beneficial when a problem seeks qualitative outputs rather than quantitative ones. Common parameters include the penalty term, the kernel type (such as linear or radial basis function), and gamma value (also known as a kernel coefficient). SVMs are well-known for assessing the

cognitive state of a driver [85–88] and addressing mental workload from a more general context [82, 89]. Classification of driver workload has also succeeded under decision trees (DTs) and k-nearest neighbors (KNNs) [90].

2.2.2.3 K-Nearest Neighbors

K-nearest neighbors (KNN) is a classification technique that seeks a majority point from k neighbors of a data point in space. In the case of binary classification under the label set $\{0, 1\}$, KNN can be defined as [91]

$$f_{KNN}(x') = \begin{cases} 1, & \sum_{i \in N_K(x')} y_i \geq 0 \\ 0, & \sum_{i \in N_K(x')} y_i < 0 \end{cases}, \tag{2.4}$$

where K is the neighborhood and $N_K(x')$ is the set of indices of the K-nearest patterns. In non-binary classification, the function can be rewritten as

$$f_{KNN}(x') = \arg\max_{y \in y} \sum_{i \in N_K(x')} I(y_i = y), \tag{2.5}$$

where $I(\cdot)$ is an indicator function that returns one if true and zero if false. Weights can be either uniform or inversely proportional to the distance from a point of interest. Other parameters include leaf size and number of neighbors. Some relevant KNN applications include cognitive stress recognition [92] and brain-computer interfaces [93].

2.2.2.4 Decision Trees

Decision trees (DTs) are a supervised learning technique that forms predictions and classifications based on decision rules established from the features of some given data [94]. It is based on the pre-computational concept of a decision tree, which is

essentially a flowchart in which every non-terminating block is a question with two possible paths. Some DT parameters include the minimum samples needed to split a node or exist at a leaf node as well as the criterion for measuring a leaf split's quality. DTs are useful due to the ease of data preparation and visualization but function at the expense of possible overfitting and unbalanced data.

2.2.2.5 Random Forest

Random forest (RF) is a type of ensemble method, that is, one that aggregates multiple classifiers. As the name forest suggests, it is an ensemble of decision trees in which some trees are given extra weight to incorrectly predicted data points [95]. This type of averaging helps in improving accuracy and reducing the possibility of overfitting.

2.3 From Human Control to Autonomy

In recent years, autonomy has aided in easing cognitive workload [8, 96]. From there, the challenge lies within applying a seamless blend of human and autonomous control based on the needs of one's mental load. The concept of sliding scale autonomy has greatly aided in this effort, in which a variable amount of autonomy can be adjusted at any given moment [97, 98]. In addition to adjusting between various levels seamlessly, the issues of trust [99] and adaptability [100] between human and autonomy must also be taken into consideration as well as deciding an appropriate amount of interaction [101]. Meanwhile, the field of unmanned aerial vehicles (UAVs) poses its own unique challenges of integrating into a shared airspace [102–104].

Most established UAV applications are primarily under remote human-centric control [25–27, 105, 106], which require many human factor considerations similarly to manned aircraft [107–111], especially when piloting multiple UAVs simultaneously

[112]. However, recent research has led to the expansion of fully autonomous control [113–118] in a variety of applications, such as search and rescue [119] and 3D mapping [120]. From here, minimizing discrepancies between predicted UAV behavior and actual outcomes is an ongoing task to ensure a safe and reliable flight. Prior solutions include the use of machine learning techniques similarly to those in Subsection 2.2.2. Other approaches involve the use of cognitive architectures [121] such as Soar [122, 123] and ACT-R [124].

2.4 UAV Autonomy in Flight

In recent years, unmanned aerial vehicles (UAVs) have become a dominant force in military operations as well as in commercial and recreational applications. UAVs can be controlled either manually via a remote pilot or autonomously. Either case presents numerous degrees of variability and unpredictability due to the behavior of the human pilot or autonomous system and can adversely affect team performance [125], leading to the necessity of blame assignment [126]. Thus, creating a mechanism for explaining the behavior and decisions of autonomous UAVs is of particular interest, due to the novel factors that comprise such an explanation. These can include memory-based data structures, memory reconstruction, explanation algorithms, and explanation interfaces.

Episodic memory is a valuable tool that can substantially aid in many of these factors [127]. It is a type of memory in which past experiences or events can be remembered explicitly [128–133], such as remembering a name and circumstances of a specific interaction [134]. From here, an autonomous agent can learn from past experiences [135]. It is particularly useful due to the ability to determine a quality of interestingness in intelligent agents [123, 136–139] as well as in other applications such as image processing [140, 141], emotional intelligence for intelligent agents [142],

or modeling of autobiographical memory [143, 144]. In a simplified context, interestingness can be expressed as a significant difference between a predicted result and an actual result. While correlating and predicting decisions based on sensory data [145] is already a fundamental and well-established concept, doing so in the context of finding interestingness to explain UAV behavior forms a new realm of possibilities.

2.5 Computational Approaches to UAV Data Prediction

Prediction of UAV-based attributes is presently a well-established field spanning a wide array of applications. Over the past two decades, these applications have steadily shifted from human-centered manual control to self-contained autonomy. For instance, under the early stages of manual UAV control, research efforts involved the assessment and prediction of human performance, such as using quantitative models for predictions regarding multi-UAV flight control [146]. Eventually, this evolved into a more supervisory role, with limited amounts of UAV autonomy used to reduce human workload [147]. Thus, prediction focused on cognitive capacity based on factors such as wait times and situational awareness.

Other early works in UAV prediction stemmed from a focus on control systems, often using common machine learning techniques to aid in a vehicle's control system. One such effort involves the development of an output feedback controller based on neural networks [148]. Controller design was improved years later as support vector regression was proposed as a direct alternative for the kinds of dynamic control systems needed for a UAV [149]. The focus on control is also present in more non-traditional UAV applications, such as in modeling a vehicle to perch on a wire in a manner similar to that of a bird for the purpose of evading attacks [150]. Accomplishing such a task requires a significant amount of creativity regarding aerodynamic modeling and

design of an effective control system, which was met through physically-based basis functions for attributes such as position, velocity, and angle.

In addition, machine learning approaches have been used in a variety of additional UAV-related fields. This includes applications based on drone images, such as tree detection from LiDAR data through a variety of algorithms [151], classification of informal settlements from 2D and 3D features via feature extraction [152], and traffic monitoring from video data by statistical profile generation [153]. Engine fault prediction is another area, leveraging particle filters to predict needs and urgency for system maintenance [154].

Because the portion of this research focused on unmanned flight safety involves location and navigation, it is appropriate to consider a major amount of related work in these two areas. Previous efforts regarding prediction of individual navigation-based attributes such as altitude or speed have in at least some capacity spanned as far back as initial stages of UAV research. An early example includes the assessment of a UAV's situational awareness, or more specifically the ability to autonomously monitor an object of interest based only on its trajectory [155]. This was accomplished through modeling of physics based differential equations as a form of state estimation. Later work involved estimating location-based qualities of the UAVs themselves, such as prediction of altitude based on a single integrated camera and a semi-supervised machine learning approach [156]. Other predictions included a UAV's heading and external wind conditions using time integration of inertial navigation system data [157] as well as estimation of a vehicle's mobility in the context of an ad hoc UAV network [112].

Using prediction for the purpose of forming entire paths is a logical next step, and prior works have addressed this manner quite extensively. These consist of basic localization problems, such as estimating a location based on digital elevation map data and infrared images. This was done in order to aid UAV navigation in mountainous

and dark terrains in which charge-coupled device (CCD) camera data are ineffective [158]. Other approaches focus on path planning, including a waypoint-based design to minimize location uncertainty [159] as well as a multi-UAV approach based on data fusion for a security-focused application of intelligence, surveillance, and reconnaissance [160]. Trajectory tracking is another method, originally used in situational awareness [147] by tracking objects relevant to a UAV. This evolved into adaptive trajectory tracking for aiding in UAV guidance [161]. Collision avoidance is another vital aspect of navigation, developments of which have been based on factors such as visual detection [162] and flight space geometry [163].

2.5.1 Sensor Fusion

Sensor fusion, or the ability to fuse dissimilar data to enhance algorithmic effectiveness, has been relatively commonplace in navigational applications. For instance, many algorithms have sought to fuse global positioning system (GPS) and inertial navigation system (INS) data to solve the problem of GPS outages in critical military and civilian applications [164–169]. This type of information fusion has also made its way to UAV navigation [170, 171]. Other relevant uses for sensor fusion have been found in more mission-specific applications, including command and control systems [172], situational awareness [173], and target identification [174]. Efforts in UAV navigation research have also improved under data fusion [157, 160] as well as applications in engine diagnostics [175, 176] and sensing networks [177].

2.5.2 Neural Network Data Fusion

While artificial neural networks are not explicitly a form of data fusion, many previous research efforts have successfully implemented ANNs alongside different data fusion techniques to address a wide variety of applications. For instance, [110] ad-

dressed multiple methods of fusion in tool condition monitoring. In this approach, sensor fusion aided in integrating feature elements, which were then trained by an ANN. The combination of tool condition monitoring, data fusion, and neural networks has been surprisingly common and has shown up in many different works over the past 30 years [110, 178, 179]. Similar applications have included fault diagnosis for manufacturing systems [180] and water quality estimation [181], while less similar fields have involved cloud analysis [182], smart cars [183], and multitemporal change analysis [184]. Regarding specific data fusion approaches, both Bayesian Inference [185] and Dempster-Shafer evidence Theory [186] have been integrated with ANNs. When considering relevance to the application of this paper, there has been work in applying ANN data fusion to target tracking [187] and localization of mobile users [188]. More relevant areas, such as estimating the location or path of a UAV, has not yet been addressed by neural network data fusion.

2.5.3 Dempster-Shafer Evidence Theory

Dempster-Shafer Evidence Theory (DSET) is a unique data fusion technique based on Bayesian combinational probability that approaches the concept of combinational factors in a distinctly different direction [189]. While Bayesian statistics utilize combinations of internal probability factors within a system [190], typically by use of random variables or propositions, DSET dynamically involves external evidence factors [191, 192] to make decisions or predict values. Each factor contains a data range in which a potential output value can lie as well as a confidence probability that the data range is reliable, or certain. DSET is meant to model information that, under more traditional Bayesian methods, would be considered incomplete [189].

2.5.3.1 Bayesian Origins

Bayesian probability is a technique in which two or more internal probabilistic factors are combined to form a single hypothesis or inference, often using random variables to fulfil unknown quantities in the context of a problem [193]. If only two factors, X and Y, are considered, then Bayesian inference can be defined as

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}, \tag{2.6}$$

where $P(X|Y)$ denotes the probability of event Y occurring given event X, $P(Y)$ represents the probability of Y alone, $P(X)$ is the probability of X individually, and $P(Y|X)$ is the probability of X given Y.

Bayesian estimation has been established in a variety of applications, including wind speed probability distribution [194] and battery lifetime analysis [195].

2.5.3.2 Theoretical Foundation

From a theoretical context, Dempster-Shafer Evidence Theory can be best described as a generalization of Bayesian probability theory, forming a divergence based on the concept of ignorance. Typical Bayesian approaches assume an ignorance quantity of zero, or equivalently 100% confidence, for any probabilistic value, which is usually appropriate when a probability factor is internal. Because DSET is based on external factors, however, this property can no longer be unconditionally assumed, and thus a plethora of evidence factors are used to determine all possible outcomes in a given scenario, each with its own degree of confidence that serves as a weight of evidence [196]. The finite set of all possible mutually exclusive values is known as the frame of discernment and is denoted by Θ. From there, the union of all subsets is given by the power set 2^{Θ}.

One of the most significant fundamental concepts of DS Theory is the basic prob-

ability assignment (BPA) function, which is essentially the equivalent of the random variable in Bayesian probability theory and is also known as the belief variable [197], denoted as m. Suppose $A_1, \ldots, A_n$ are all possible sets within a frame of discernment, noting that $A_i \in 2^\Theta$. Then

$$m : 2^\Theta \to [0,1], \sum_{i=1}^{n} m(A_i) = 1, m(\emptyset) = 0, \tag{2.7}$$

where $\emptyset$ denotes the empty set. Each set A_i takes the form of the interval ($[\underline{x}, \bar{x}]$), denoting the lower and upper bounds of an evidence factor, respectively [197]. Each BPA m consists of two vital properties: a range, as established by a set A, and a degree of confidence μ that lies within the range of [0,1] [198]. Thus, ignorance can be defined as simply $1 - \mu$. At certain times, manually adding new BPAs can be an exhaustive process, in which case sampling from a distribution becomes an ideal solution. This process, also known as discretization, involves generating n discrete samples based on a lesser amount of initialized BPAs, which collectively form a cumulative density function (CDF) of BPA structures [199].

Once two or more evidence factors from different sources exist in the same problem, then typically the aggregation of BPAs becomes essential [197]. Many possible methods exist in which this task can be accomplished, but the scope of this research focuses primarily on Dempster's rule of combination. Suppose m_1 and m_2 denote two BPA functions that are to be aggregated. Under Dempster's rule, a new BPA function $m_1 \oplus m_2 : 2^\Theta \to [0,1]$ is defined as [200]

$$[m_1 \oplus m_2] = \frac{\sum_{A \cap B = y} m_1(A) m_2(B)}{1 - \sum_{A \cap B = y} m_1(A) m_2(B)} \tag{2.8}$$

on the condition that $y \neq 0$. Otherwise $[m_1 \oplus m_2] = 0$.

Belief and Plausibility are two of the most vital core functions of an aggregate

BPA, as they represent the best and worst case scenarios, respectively, in a frame of discernment and are defined as

$$Bel(B) = \sum_{A_i \subset B} m(A_i) \tag{2.9}$$

and

$$Pl(B) = 1 - \sum_{A_i \cap B = \emptyset} m(A_i), \tag{2.10}$$

where $m(A_i)$ denotes the BPA mass function for a set A_i.

Some applications that have incorporated DSET in the past include a variety of localization problems, such as in wireless sensor networks [201], robot localization [202], and indoor localization of mobile users [203]. Other applications include image classification [204] and detection of credit card fraud [205]. As previously mentioned, it has been implemented successfully in fusion of sensors for enhanced vehicular navigation [166, 168]. However, this framework has not been implemented in UAV-based data prediction.

2.5.4 Kalman Filters

When discussing prior work involving UAV data prediction, an important class of algorithms is that of the Kalman filter. Kalman Filter is a type of parameter estimation in which mathematical models can be formed based on measured data [206]. It is based on a two-step Bayesian model of stochastic filtering that consists of prediction and correction. However, it has also been shown that these filters tend to underperform ANN-based solutions in related applications [207].

2.5.4.1 Extended Kalman Filter

The Extended Kalman Filter (EKF) is a variation that implements instantaneous linearization at each time step to approximate nonlinearities [206]. It is considered the

most popular method of nonlinear filtering in the field of aerospace. More specifically, it has been applied in UAV localization [208] and trajectory prediction [209].

2.5.4.2 Unscented Kalman Filter

The Unscented Kalman filter (UKF) differs from EKF in that less linearization is involved [206]. Instead, it uses sample points to approximate a probability distribution. UKF is not quite as widely used as EKF in this area. However, it does have established roots in UAV navigation [210] and general parameter estimation [206].

Chapter 3

Cognitive Workload Assessment

In order to address human-side flight performance, the concept of cognitive workload becomes a subject of particular importance. This chapter details the portion of the dissertation research concerning the assessment of cognitive workload and bridges it with fundamental machine learning (ML) concepts. This includes a conceptual review of both cognitive load and its previous applications in machine learning as well as the details of the experimental procedure to compare and contrast various candidate ML techniques on multiple sets of cognitive data. Following thorough experimental results, an extensive cost-benefit analysis of the different fundamental methods is then discussed, followed by conclusions regarding ML's potential roles in human-oriented flight performance.

3.1 Introduction

As discussed in Chapter 2, cognitive workload is the amount of mental effort that is stored in working memory. It can be perceived as a balancing act between overload (e.g. driving in downtown Chicago traffic while talking to passengers, using a smartphone, and listening to the radio) and underload (e.g. driving on a rural highway with other no vehicles around, no passengers, and the radio turned off). In the field of flight safety, overload is typically the more common of the two extremes. The use of

Table 3.1: Summary of cognitive workload datasets used in this experiment.

ML Technique	Subjective Parameter	Iterative Parameter 1	Iterative Parameter 2
Artificial neural network (ANN)	Training (SGD, LBGFS)	No. hidden layers	No. hidden neurons
K-nearest neighbors (KNN)	Weights (uniform, distance)	No. neighbors	Leaf size
Support vector machines (SVM)	Kernel (linear, RBF)	Penalty term	Gamma value
Decision tree	Criterion (gini, entropy)	Min. samples for split	Min. samples for leaf
Random forest	Criterion (gini, entropy)	Min. samples for split	Min. samples for leaf

sliding scale autonomy is a common way to reduce a pilot's workload, while underload could be resolved with a simple alert. In these cases, it is useful to detect the potential of cognitive overload or underload as early as possible. Thus, machine learning can be applied to examine any non-invasive physiological data combined with current levels of cognitive load and thereby predict future load levels based on ongoing trends.

From there, the challenge is to identify which machine learning method is best. Many different techniques have been established in these kinds of applications, including artificial neural networks (ANNs) [41, 43, 58, 73, 74, 79, 80], support vector machines (SVMs) [82, 85–89], k-nearest neighbors (KNNs) [90, 92, 93], decision trees (DTs) [90], and random forests (RF) [95]. Thus, this phase of the research consists of a twofold approach: compare and contrast the established techniques based on extensive experimental analysis, and develop an adaptive algorithm that dynamically selects a technique based on the attributes of some given physiological data. This chapter focuses on the former.

When performing an analysis of alternatives, it is important to have as many relevant variables as can be reasonably accommodated. One of the most apparent is a quantity and variety of data. This chapter approaches data variability by starting with more generalized physiological data with a sufficient amount of relevance to cognitive workload, then evolving to a more specific dataset that directly involves pilots and flight safety. The remaining variables remain specific to the different machine learning techniques to be analyzed. These attributes can be subjective, such as ANN training

method, SVM kernel, KNN weight considerations, and DT/RF criteria. They can also be more iterative in nature, such as the number hidden layers or hidden neurons in an ANN, penalty term and gamma value in a SVM, number of neighbors and leaf size in KNN, and minimum number of samples for split or leaf in DT and RF. Table 3.1 provides an overview of the three method-specific parameters to be selected for each technique.

By analyzing all these candidate machine learning techniques under the aforementioned experimental parameters, the advantages and disadvantages of these methods under the context of assessing cognitive workload can be realized in determining selection of the most beneficial method at any given point in time based on the attributes of any given physiological data.

3.2 Methods and Materials

A generalized flowchart of the analysis of alternatives process is presented in Fig. 3-1. The initial step is to select the dataset to be tested. A description of the datasets used is provided in the next section. Next is to select a machine learning technique to be evaluated. In doing so, three method-specific parameters are compared: two of which are numerical and iterative, and the third of which is more subjective and typically consists of two options. The following step is to evaluate the method based on fundamental metrics such as accuracy and computational runtime. Other metrics for consideration include F1 score, precision, and recall. In viewing and analyzing the results of these metrics, the final step is to determine the pros and cons of the given machine learning method.

From there, a more comprehensive approach is one that is iterative, traversing each dataset, method, and method-specific parameter. This approach is detailed in Fig. 3-2. Due to the extensive use of *for* loops, the entire process is automatic, inputting the

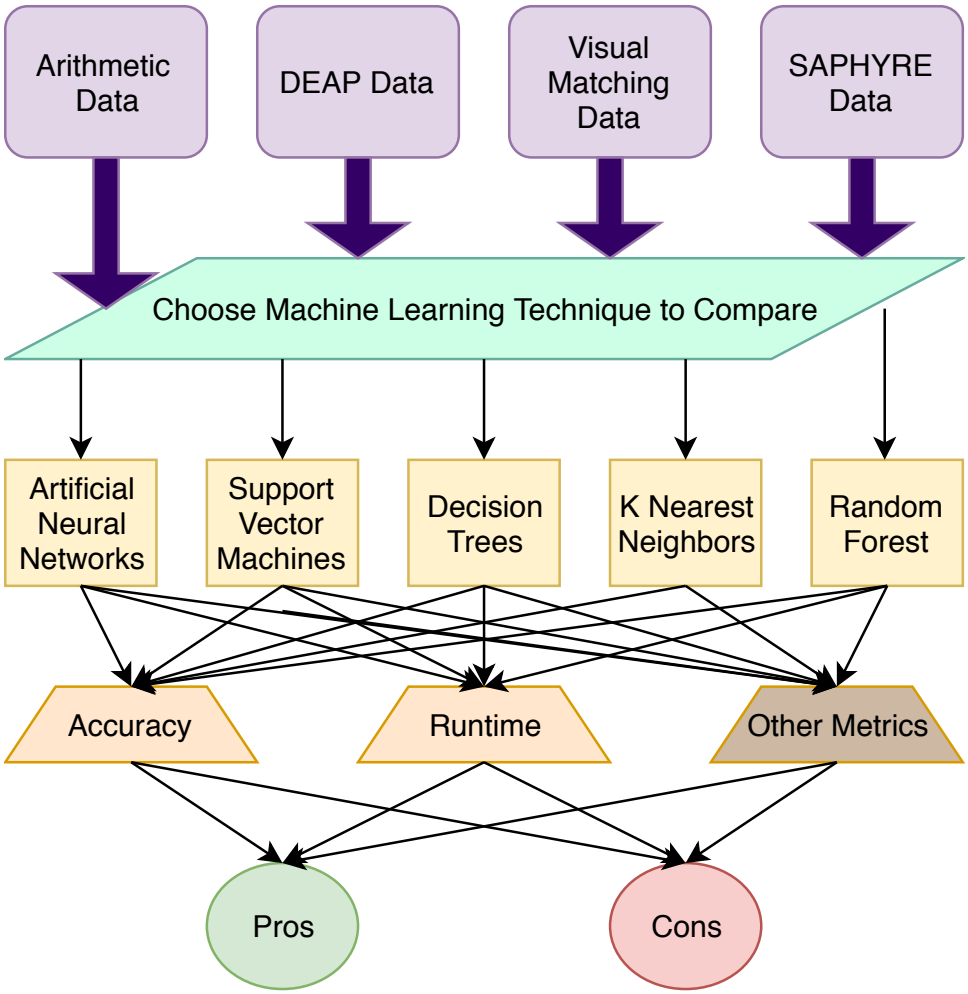


Figure 3-1: Flowchart of experimental process.

formatted input and output files of each dataset in comma separated value (CSV) format and outputting the results in both numerical and graphical formats. The former consists of a CSV file for each combination of dataset, method, subjective parameter (e.g. ANN training technique, SVM kernel), and type of evaluation metric. In each file is the numerical result for the selected metric for each iterative parameter combination. The latter format consists of a PDF file containing a three-dimensional surface plot, again for each combination of dataset, method, subjective parameter, and type of evaluation metric. The details of these graphical results are described in Section 3.3.

The overall process is written in Python and incorporates the scikit-learn package

```
for all datasets do
  train_in, val_in, train_out, val_out ← all possible input and output data
  for all methods do
    for all subjective parameters do
      for all trials do
        for first iterative parameter do
          for second iterative parameter do
            create classifier using selected method and parameters
            train classifier
            results ← [train. error, val. error, precision, recall, F1, runtime]
            for all results do
              result ← result / no_iterations
            end for
          end for
        end for
        for all results do
          save numerical result values for all iterative parameter combinations
          save surface plot of values for all iterative parameter combinations
        end for
      end for
    end for
  end for
end for
```

Figure 3-2: Pseudocode of complete experimental process.

Table 3.2: Summary of cognitive workload datasets used in this experiment.

Name	Description	Relevance	Source
Arithmetic	Fear conditioning and cognitive load	EDA resultant from cognitive tasks	co-author of [67]
DEAP	Human affective states	EEG recorded from watching video clips	[212]
Visual Matching	Genetic predisposition to alcoholism	EEG recorded from observing two images	[213]
SAPHYRE	Pilot skill level	Pilot workload based on heart rate	co-authors of [214]

[211] for access to built-in machine learning functions. The experiment was performed using the Anaconda Python environment on a Windows 10 desktop with an Intel Core i5-3570 3.40 GHz quad core CPU and 16 GB of RAM. In addition, the results were presented as the averages of 10 independent trials for redundancy and consistency.

3.2.1 Data Acquisition

Table 3.2 provides an overview of the four datasets obtained for this research. The second and third sets were obtained from the public domain, while the first and fourth were from a collaborator described in Acknowledgements. Each dataset involves a classification problem of some set of cognitive activities. For instance, adding two repeatedly or subtracting seven repeatedly requires increasingly high amounts of mental load, and thus the goal here is to determine which activity took place at a given time step given the physiological data values at that time and accurately classify which activity (including which specific iteration of add two or subtract seven) took place. By doing so and by interpreting the amount of cognitive load anticipated in each activity, meaningful workload assessment can take place.

3.2.1.1 Arithmetic Dataset

The first dataset utilized in this research is based on an experiment by [14] in 2016. Natarajan et al. collected the data for the purposes of evaluating fear conditioning and cognitive load. In turn, the data has been formatted so that it focuses entirely on the latter. To address cognitive workload, the data measures electrodermal activity

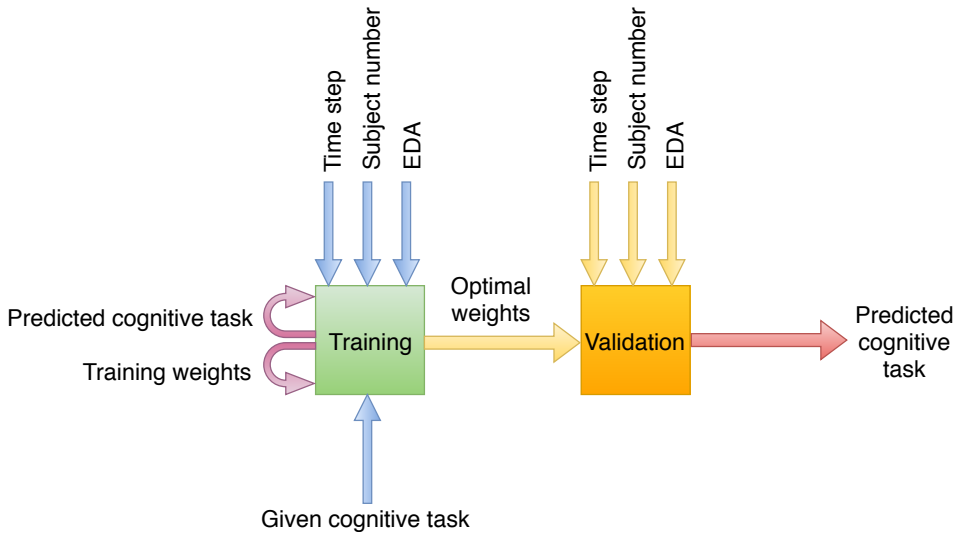


Figure 3-3: Supervised machine learning process for arithmetic data.

(EDA) during a series of tests in which a human subject is required to either add 2, subject 7, or rest for 30-second periods. These mental tasks were chosen for the specific purpose of assessing cognitive workload, based on an individual's ability to remember a number and repeatedly, as well as interchangeably, add to or subtract from it. For this research, the data was repurposed so that an ANN or SVM could be modeled to answer one of many possible questions:

1. Given a task number and a time step, what will be a subject's EDA?
2. Given a subject's EDA and a time step, which task was he/she performing?
3. *Given a subject number, his/her EDA, and a time step, what task was he/she performing?*

The data in this research is formatted to focus on addressing the third and italicized question. The three inputs are self-explanatory as given in the third question, while the sole output is the activity number, which represents the task being performed. Although there are only three unique activities, each nonconsecutive instance of a recurring non-resting task is its own separate activity number for a total of five

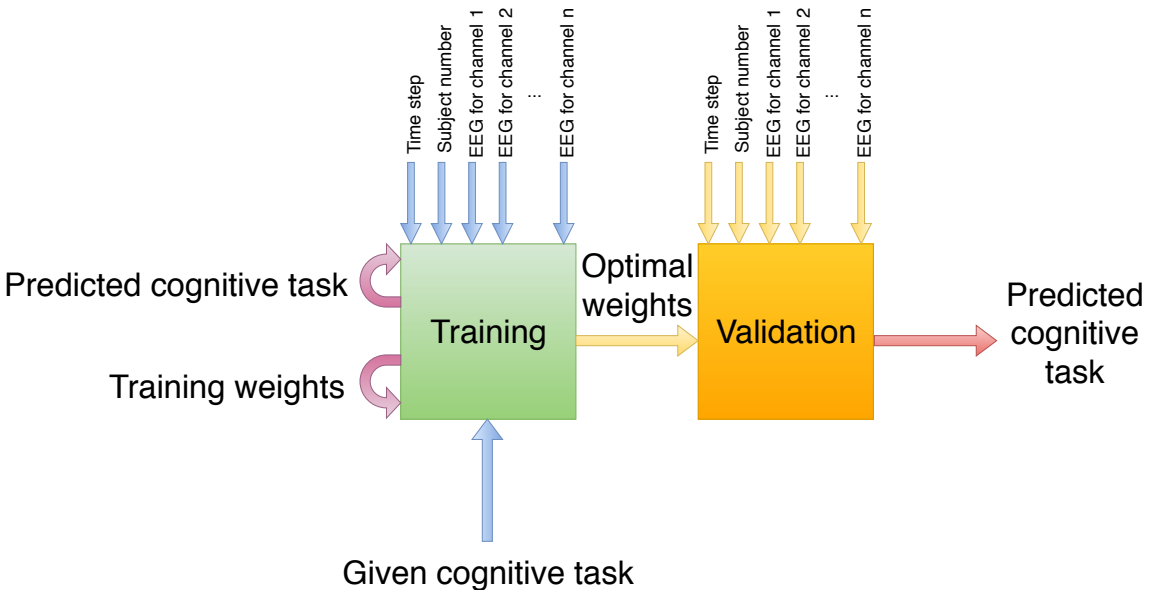


Figure 3-4: Supervised machine learning process for DEAP data.

possible classifications. Figure 3-3 presents the overall machine learning process for this data.

3.2.1.2 DEAP Emotion Recognition Dataset

The second dataset named Database for Emotion Analysis using Physiological Signals (DEAP) was obtained from [212] and examines emotional response to different music videos. Given the sheer amount of visual creativity involved in many music videos, it can be established that the amount of cognitive load required to absorb such videos can vary rapidly. The inputs consist of EEG in 40 channels as well as time step and subject number, while the output is simply the video being watched on the condition that each video has a varying level of cognitive load required to adequately absorb what is being watched. Therefore, this data can be modeled based on the question: *Given a time step, a subject number, and the subject's EEG, what task was he or she performing?* Fig. 3-4 provides an overview of this dataset's machine learning process as well as its inputs and outputs.

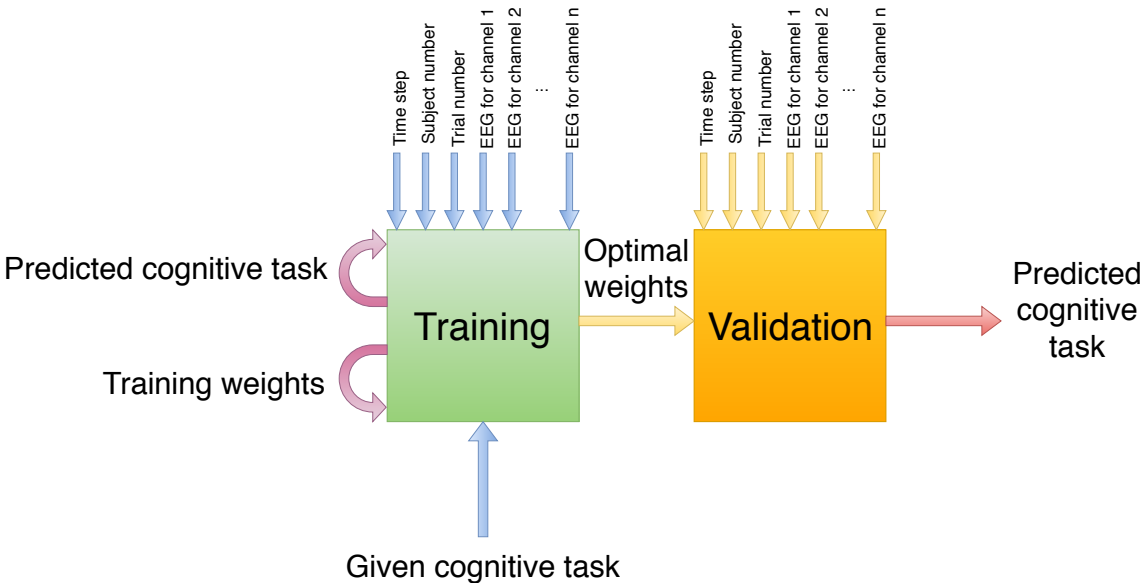


Figure 3-5: Supervised machine learning process for visual matching data.

3.2.1.3 EEG Visual Matching Dataset

This third dataset was obtained from [213], which examines genetic predisposition among alcoholic and control subjects based on EEG correlation. Over the course of 122 subjects and 120 trials, the cognitive activity consisted of being shown either a single image, two matching images, or two non-matching images. Each of these three activities was considered as separate classifications between alcoholic subjects and control subjects, hence a total of six possible output choices. Thus, this classification problem is modeled from the modified question: *Given a time step, a subject number, a trial number, and the subject's EEG across 64 channels, what task was he or she performing?* Fig. 3-5 depicts an overview of this data's machine learning process as well as the inputs and outputs.

3.2.1.4 SAPHYRE Pilot Dataset

The final dataset bears the most relevance to the subject of flight safety, as the human subjects are actual pilots. In addition, this data consists of both physiological inputs and navigational/operational inputs. Due to the large amount of unique inputs,

there is no flowchart provided here. Instead the inputs are listed as follows:

1. Time step
2. Subject number
3. Heart rate
4. Aileron
5. Elevator
6. Rudder
7. Throttle
8. Heading
9. Altitude
10. Latitude
11. Longitude
12. Speed
13. Proposed latitude
14. Proposed longitude
15. Distance from path

From there, the output classification is a combination of flight route, flight path, and pilot skill level, which is given by

$$C = 5sp + r, \tag{3.1}$$

where C is the output classification, s denotes a pilot's skill level (1 for expert, 0 for novice), p is the path number, and r represents the route number. This is on the basis that cognitive workload levels could vary with skill level. In other words, a more experienced pilot may need less mental load to complete the same task undertaken by a novice pilot. This is also based on the fact that the complexity of different routes or paths could also contain varying levels of cognitive load. Thus, the overall problem given by this data models the question: *Given the above inputs, what are the route, path, and pilot skill level?*

3.2.2 Evaluation Metrics

To ensure a thorough and meaningful analysis of the experimental results, five different metrics were selected for evaluating the machine learning techniques, with the majority of importance lying within accuracy and runtime.

In scikit-learn, accuracy is determined by the score function, which in each classifier is the mean accuracy. Mean accuracy is simply the ratio of correct predictions to total possible predictions. This is considered to be a tough evaluation metric, as a classification is only considered correct if there is an exact match between the predicted output and the actual output, as opposed to accuracy in a regression problem, in which evaluation is based on the closeness between a prediction and an output. Because each of the candidate machine learning methods are all supervised learning techniques, accuracy in this experiment is divided into training accuracy and validation accuracy, in which the training samples and validation samples are respectively evaluated individually.

Computational runtime is the amount of time required to complete the training and evaluation phases for each combination of dataset, method, and other parameters. Runtime is dependent on both efficiency of the ML method and the computing resources available. Thus, every facet of this experiment is done on the same hardware

and software, the specifications of which are discussed at the beginning of Section 3.2.

Other relevant evaluation metrics include precision, which differs from accuracy in that it is the ability of a classifier to not predict a positive output for a sample with a negative one [211]. It is defined as the ratio of number of true positives to the sum of both true and false positives. Recall, however, is the ability to find all the positive samples and is calculated as the ratio of true positives to the true positives and false negatives. Meanwhile, the F1 score is a weighted average of precision and recall, defined by

$$F1 = 2 * \frac{precision * recall}{precision + recall}. \tag{3.2}$$

With exception of runtime, all of the evaluation scores are normalized values, in which 1 is a perfect score and 0 is the lowest possible.

3.3 Experimental Results

Extensive results of the entire experimental procedure are given in the following subsections, figures, and tables. For organizational purposes, each subsection corresponds to each of the four datasets. Within each subsection are two tables that summarizes the results for each combination of technique and subjective parameter (e.g. ANN optimizer, SVM kernel). The result types consist of training accuracy, validation accuracy, and runtime (in seconds) in the first table as well as precision, recall, and F1 scores in the second table, with each category consisting of a minimum value, a maximum value, and a mean value.

Each subsection also consists of five figures, one for each ML method, with each figure divided into 3-by-2 subfigures. Each column corresponds to the subjective parameter of the method (such as gini and entropy criteria for DT and RF), while each row corresponds to the type of evaluation metric presented. In this case, the focus is only only three types: validation accuracy, F1 score, and runtime. Each

Table 3.3: Summary of training accuracy, validation accuracy, and computational runtime on arithmetic data.

	Train. Accuracy			Val. Accuracy			Runtime (s)		
Technique	Max	Min	Mean	Max	Min	Mean	Max	Min	Mean
ANN SGD	.990	.405	.707	.990	.403	.706	24.8	.995	8.68
ANN L-BFGS	.999	.399	.704	.998	.393	.703	43.1	.229	10.8
KNN Uniform	1.00	.988	.994	.991	.980	.985	.473	.287	.348
KNN Distance	1.00	1.00	1.00	.991	.987	.989	.482	.297	.353
SVM Linear	.437	.435	.436	.442	.431	.436	7.73	6.63	7.21
SVM RBF	.993	.892	.983	.987	.891	.977	10.1	2.70	4.38
DT Gini	1.00	.999	.999	.999	.998	.999	.225	.203	.210
DT Entropy	1.00	.999	.999	.999	.998	.999	.230	.205	.216
RF Gini	1.00	.998	.999	.998	.997	.998	.331	.310	.318
RF Entropy	1.00	.998	.999	.998	.996	.997	.405	.356	.368

subfigure is a three-dimensional surface plot with x and y axes corresponding to the numerical, or iterative experimental parameters, while the z axis is the evaluation result.

3.3.1 Arithmetic Dataset

Table 3.3 provides an overview of the accuracy and runtime results for the arithmetic dataset. In terms of accuracy alone, nine out of ten technique combinations show highly promising results for this dataset, as each one other than SVM with the linear kernel provides up to at least 99% accuracy for both training and validation. In addition, DT, RF, and KNN provide perfect scores in training accuracy in some scenarios. In the case of the latter, every combination of iterative parameters results in 100% accuracy. In terms of ranges of values, some methods are better at producing high accuracy across the board than others. In the case of DT and RF, the lowest accuracy value produced is still over 99%, with the former having a slight edge in

Table 3.4: Summary of F1, precision, and recall scores on arithmetic data.

	F1			Precision			Recall		
Technique	Max	Min	Mean	Max	Min	Mean	Max	Min	Mean
ANN SGD	.990	.121	.567	.992	.087	.555	.992	.200	.609
ANN L-BFGS	.998	.121	.570	.998	.086	.557	.998	.200	.612
KNN Uniform	.991	.981	.986	.994	.984	.989	.991	.976	.984
KNN Distance	.992	.988	.990	.992	.987	.990	.991	.987	.990
SVM Linear	.120	.123	.121	.088	.086	.087	.200	.200	.200
SVM RBF	.988	.901	.979	.987	.871	.973	.990	.950	.986
DT Gini	.999	.998	.999	.999	.998	.999	.999	.998	.999
DT Entropy	.999	.998	.999	.999	.998	.999	.999	.998	.999
RF Gini	.998	.997	.998	.999	.997	.998	.998	.997	.998
RF Entropy	.998	.996	.997	.998	.996	.997	.998	.997	.998

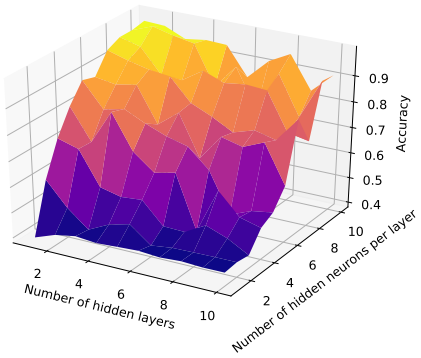
minimum and average accuracy values. ANN, meanwhile, produces the widest range of accuracy values, ranging anywhere from 39.9% to 99.8%. This is particularly true for the L-BFGS optimizer, as both the upper and lower limits are produced by this parameter. In terms of computational runtime, DT continues to maintain supremacy in this dataset, with maximum runtime values that rival even the minimum values from other methods. ANN and SVM appear to do the worst in terms of runtime and efficiency, as all their runtime values are the only ones to exceed half a second, often by an overwhelming margin. SVM has the most consistently high runtime, as every value exceeds one second, while ANN has the greatest maximum value as well as the widest range of values, spanning roughly from a quarter of a second to three quarters of a minute. Overall, it can be declared so far that decision trees are the most optimal method for the arithmetic dataset from all positive angles.

Table 3.4 is an overview for the remaining three metrics: the precision, recall, and F1 scores. From here, DT is still the most highly scored method, as each set of metric values ranges from .998 to .999. RF comes in at a close second with only

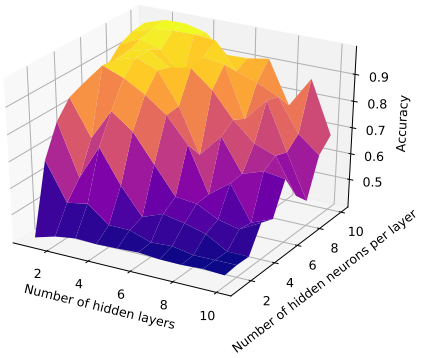
slightly lower values. ANN again produces the widest range of values in all three cases, while SVM's linear kernel continues to produce consistently poor results. Thus, DT is still the most superior method for this dataset. In terms of non-runtime metrics, both gini and entropy criteria produce the same ranges of scores. Thus, the only remaining way to compare the two is with runtime, in which case gini produces slightly faster speed and efficiency, thereby making DT with gini the best method for this dataset.

Fig. 3-6 provides graphical results for ANN on the Natajan dataset. From here, the patterns and trends in metric variability are more visible. As seen in Subfig. 3-6a, accuracy increases in proportion to the number of hidden neurons in the neural network. In terms of number of hidden layers, however, there does not seem to be any significant correlation. For Subfig. 3-6b, both axes follow noticeable patterns. The number of neurons still maintains a direct correlation with accuracy while an increase in the number of hidden layers actually decreases the accuracy. The plots of F1 scores in Subfig. 3-6c and 3-6d follow the same trends and share similar values overall. The runtime results in Subfigs. 3-6e and 3-6f follow uniquely different trends, even from one another. In both cases, an increase in the number of hidden neurons results in a greater runtime. However, in terms of number of hidden layers, runtime actually decreases in SGD in proportion to number of layers increasing, while L-BFGS runtime increases. Both of these trends, meanwhile are at near exponential rates, as seen in the curves respective to the x axis. This is again due to the wide range of runtime values produced by the ANN.

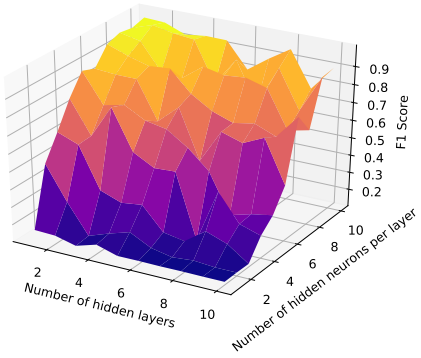
When turning to SVM results depicted in Fig. 3-7, it can be observed that under the linear kernel, there no noticeable correlations relating validation accuracy or F1 score with penalty term or gamma value. This is primarily because the differences in accuracy values are highly minimal (.008 or 0.8% for accuracy, 0.002 for F1). It is also worth recalling that this is the SVM kernel and only method configuration for this dataset in which accuracy is consistently low (in this case, never exceeding 50%).



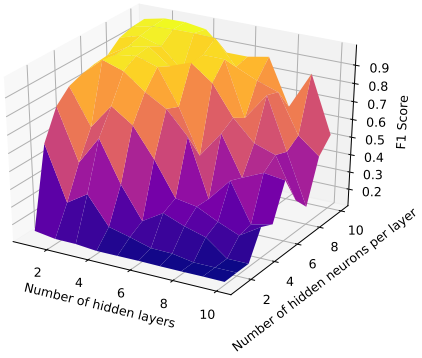
(a) Validation error for L-BFGS optimizer



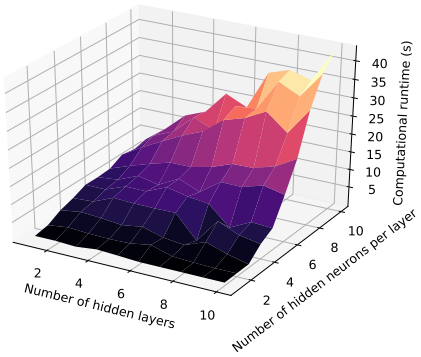
(b) Validation error for SGD optimizer



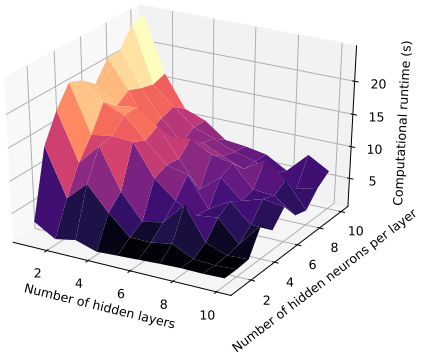
(c) F1 score for L-BFGS optimizer



(d) F1 score for SGD optimizer

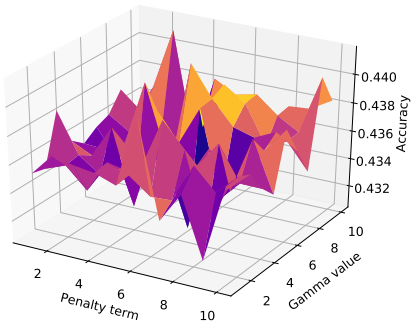


(e) Computational runtime for L-BFGS optimizer

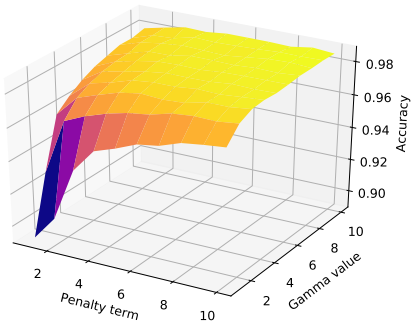


(f) Computational runtime for SGD optimizer

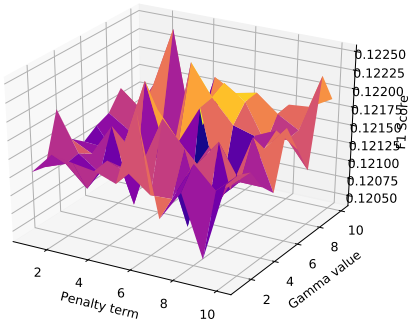
Figure 3-6: Plots of validation accuracy, F1 score, and computational runtime for ANN on arithmetic data for L-BFGS and SGD optimizers.



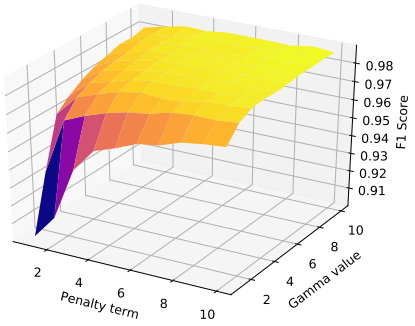
(a) Validation error for linear kernel



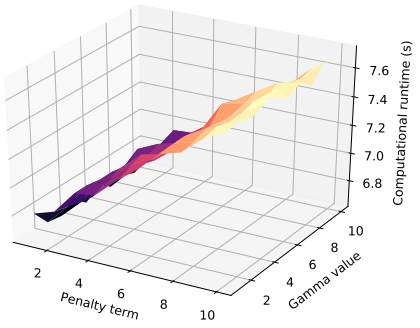
(b) Validation error for RBF kernel



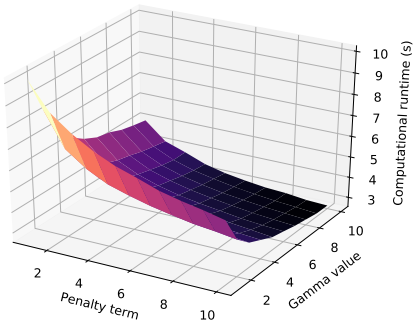
(c) F1 score for linear kernel



(d) F1 score for RBF kernel



(e) Computational runtime for linear kernel



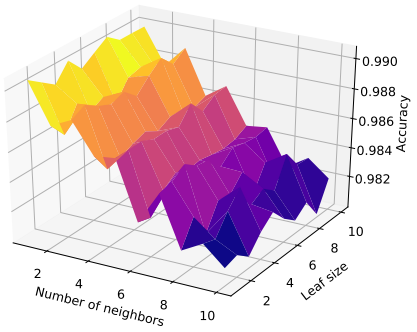
(f) Computational runtime for linear kernel

Figure 3-7: Plots of validation accuracy, F1 score, and computational runtime for SVM on arithmetic data for linear and RBF kernels.

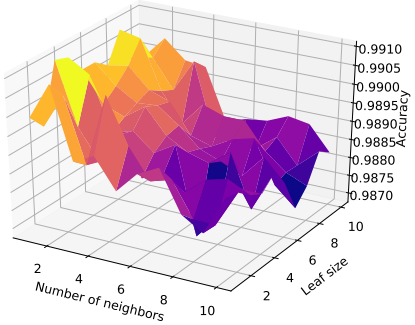
F1 score is also extremely low in this case, never exceeding an eighth of a perfect score. RBF accuracy, however, follows more predictable trends with higher values. In both axes, accuracy and F1 increase proportionally to both iterative variables, but primarily at lower values. When both axis values are four or more, the curve becomes relatively flat. As for runtime, both kernels have distinctly different effects on this metric. In the linear kernel, runtime increases with penalty term with no noticeable change with gamma value, while in RBF, the runtime is inversely proportional to either variable.

A similar set of graphical results for KNN is presented in Fig. 3-8. In the case of uniform weights, validation error and F1 score seem to decrease as the number of neighbors increases, while there is no noticeable change as leaf size changes. For distance weights, both axes have an inverse proportionality with accuracy, although it is less noticeable in leaf size. For runtime, both weight configurations involve increased runtime as the number of neighbors increases and as the leaf size decreases. It is also worth noting that these slopes are much less steep compared to the ANN results, as the ranges of values are much closer together. More specifically, all accuracy and F1 values range from .98 to .99, while all runtime values are within 0.15 seconds of one another.

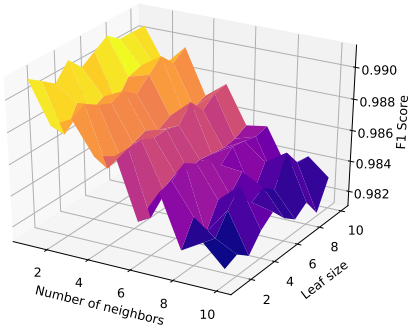
Lastly, Figs. 3-14 and 3-15 provide the results for DT and RF, respectively. In both cases, accuracy and F1 score exceed .996 (or 99.6%) under every variable combination. While there is very little variability in accuracy, it can be seen that accuracy varies proportionally with the minimum number of samples required for leaf under DT and slightly decreases with both variables under RF. Runtime varies by no more than 0.2 seconds and follows no noticeable patterns.



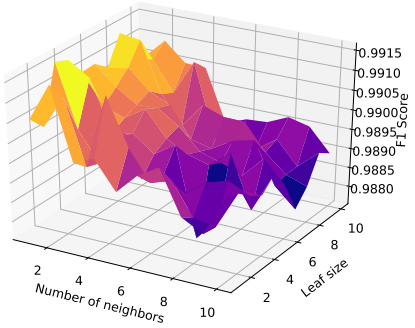
(a) Validation error for uniform weights



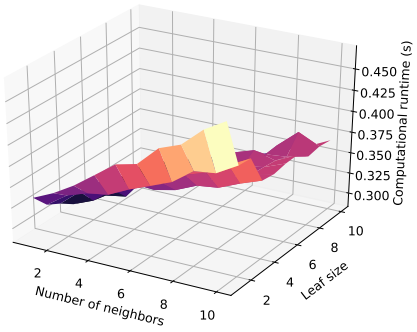
(b) Validation error for distance weights



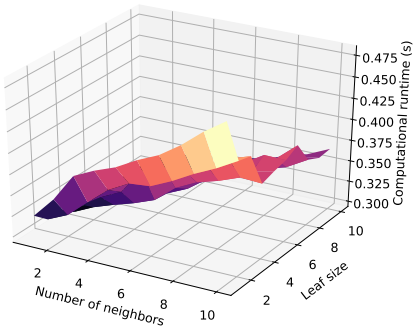
(c) F1 score for uniform weights



(d) F1 score for distance weights

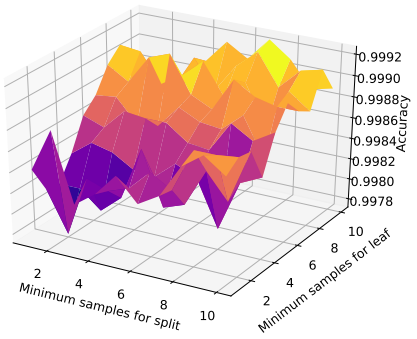


(e) Computational runtime for uniform weights

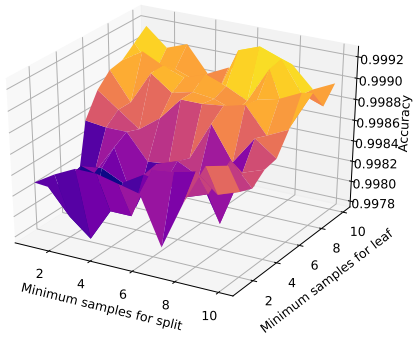


(f) Computational runtime for distance weights

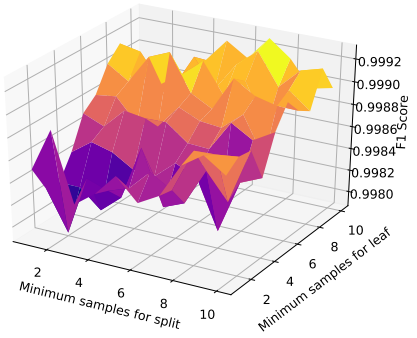
Figure 3-8: Plots of validation accuracy, F1 score, and computational runtime for KNN on arithmetic data for uniform and distance weights.



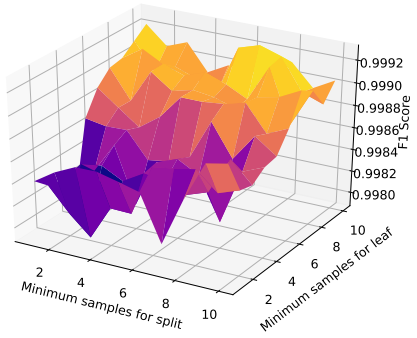
(a) Validation error for gini criterion



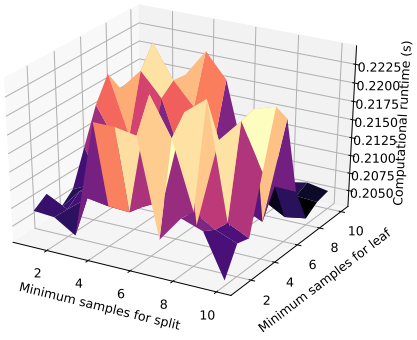
(b) Validation error for entropy criterion



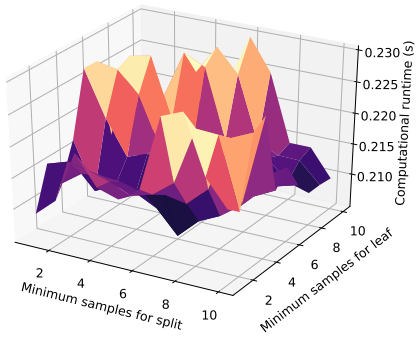
(c) F1 score for gini criterion



(d) F1 score for entropy criterion

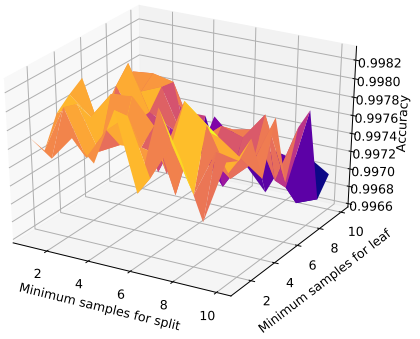


(e) Computational runtime for gini criterion

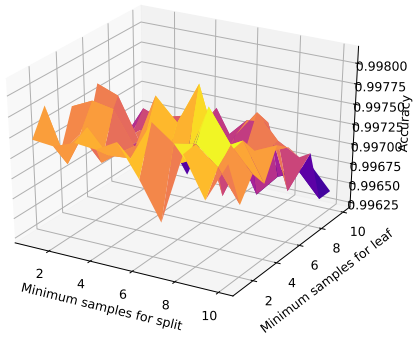


(f) Computational runtime for entropy criterion

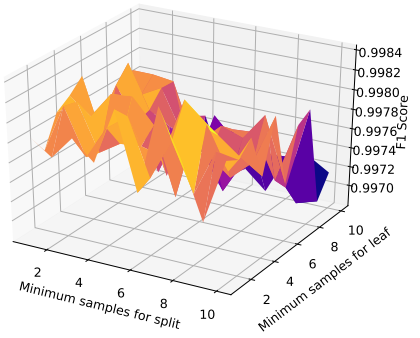
Figure 3-9: Plots of validation accuracy, F1 score, and computational runtime for DT on arithmetic data for gini and entropy criteria.



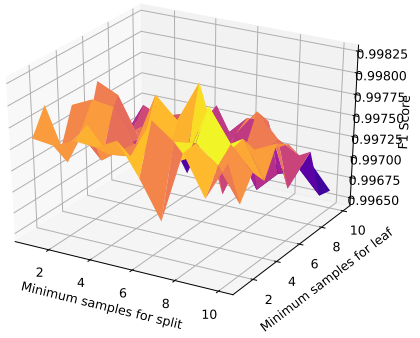
(a) Validation error for gini criterion



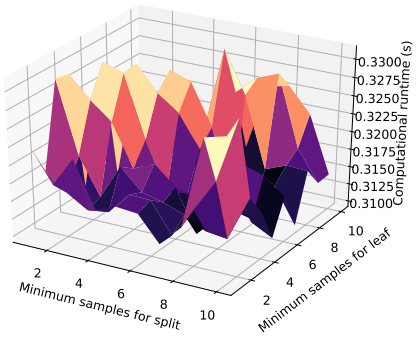
(b) Validation error for entropy criterion



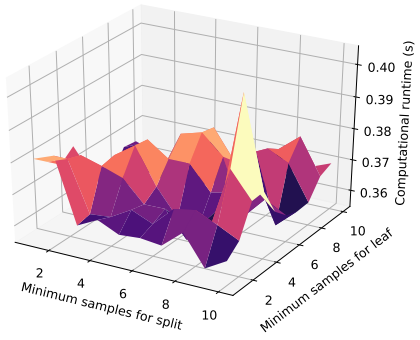
(c) F1 score for gini criterion



(d) F1 score for entropy criterion



(e) Computational runtime for gini criterion



(f) Computational runtime for entropy criterion

Figure 3-10: Plots of validation accuracy, F1 score, and computational runtime for RF on arithmetic data for gini and entropy criteria.

Table 3.5: Summary of training accuracy, validation accuracy, and computational runtime on DEAP data.

	Train. Accuracy			Val. Accuracy			Runtime (s)		
Technique	Max	Min	Mean	Max	Min	Mean	Max	Min	Mean
ANN SGD	.726	.723	.729	.729	.719	.724	8.86	4.35	6.45
ANN L-BFGS	.740	.724	.729	.728	.716	.722	66.3	3.12	18.1
KNN Uniform	1.00	.340	.475	.116	.144	.151	135	24.2	64.5
KNN Distance	1.00	1.00	1.00	.165	.154	.160	116	15.4	50.2
SVM Linear	.138	.114	.130	.104	.089	.098	55.96	52.68	54.08
SVM RBF	.982	.217	.785	.174	.129	.165	92.7	70.5	84.9
DT Gini	1.00	.491	.642	.179	.170	.175	19.8	13.6	16.6
DT Entropy	1.00	.461	.629	.183	.172	.178	24.8	18.7	21.6
RF Gini	.990	.652	.802	.180	.169	.174	21.8	15.5	18.3
RF Entropy	.990	.658	.805	.178	.165	.173	24.5	18.8	21.4

3.3.2 DEAP Emotion Recognition Dataset

Table 3.5 provides an overview of the accuracy and runtime results for the DEAP dataset in a manner similar to that of Table 3.3. A key initial observation here is that no method under these experimental constraints is able to model the DEAP data as well as the arithmetic data. ANN is the only method with remotely acceptable results, with values ranging from 71.6% to 72.9%. With exception of the SVM linear kernel, which produced low training and validation accuracy just as it did in the previous dataset, all the remaining methods have been subjected to extreme cases of overfitting. This is evidenced by the high levels of training accuracy observed in many cases, which translated poorly into the validation stage. Another key observation is that runtime for this dataset was significantly longer than the previous one, which is most likely due to the increased number of inputs involved in this data. Hence, the minimum number of seconds required for an algorithmic instance is three seconds, while some maximums are as high as one-and-a-half minutes. Much like before,

Table 3.6: Summary of F1, precision, and recall scores on DEAP data.

	F1			Precision			Recall		
Technique	Max	Min	Mean	Max	Min	Mean	Max	Min	Mean
ANN SGD	.284	.279	.281	.435	.240	.283	.334	.333	.334
ANN L-BFGS	.341	.279	.305	.415	.240	.349	.358	.333	.342
KNN Uniform	.162	.134	.145	.164	.144	.156	.162	.144	.151
KNN Distance	.164	.153	.159	.165	.154	.160	.165	.153	.160
SVM Linear	.079	.076	.090	.110	.091	.102	.103	.089	.098
SVM RBF	.173	.123	.163	.175	.140	.166	.174	.130	.165
DT Gini	.178	.169	.173	.180	.171	.175	.179	.171	.175
DT Entropy	.182	.170	.177	.183	.172	.178	.183	.172	.178
RF Gini	.179	.167	.173	.180	.169	.174	.180	.169	.174
RF Entropy	.176	.161	.171	.178	.167	.173	.178	.165	.173

SVM consistently produces the highest amount of runtime, while ANN has the most variability. However, the latter case only holds true for the L-BFGS optimizer. In SGD, runtime is consistently the lowest while validation accuracy remains the highest. Thus, conclusions can be drawn so far that ANN under the SGD optimizer is the best choice out of the methods given for modeling the DEAP dataset.

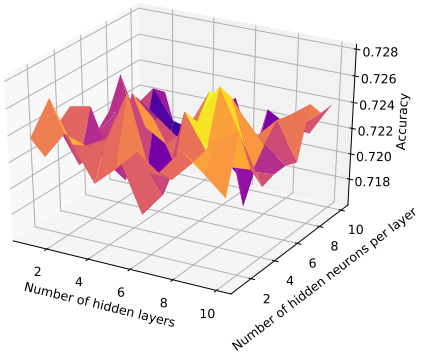
A glance of the results under the remaining metrics is given in Table 3.6. In a similar observation to that of the previous table, no method produces a particularly stellar score when compared to the arithmetic dataset. This includes ANN, although it still contains the highest values. Unlike validation accuracy, however, in which the method achieved modest values of over 70%, its highest scores are no more than 0.435. It should be noted that the values of these metrics for all other methods are consistently lower than 0.2. Thus, if stellar modeling of the DEAP dataset was a higher priority in this research, better care would have to be taken to tune parameters and modify one or more of these methods to form more accurate and more effective classifications. Because the overall focus is on flight safety, however, doing so is of

lesser importance and thus simply serves as a basis of comparison for the remainder of the experiment.

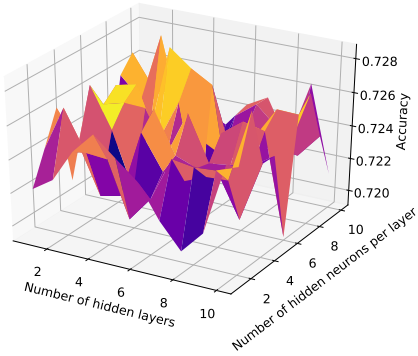
Upon taking a closer look at the results in a manner similar to that of the previous subsection, Fig. 3-11 presents graphical results of the ANN on the DEAP dataset. The most significant characteristic here is the fact that the validation error does not bear any resemblance to the F1 score, as was the case in the previous dataset. Here, the accuracy simply oscillates across a narrow range of values, while the F1 score forms a smoother surface. Under the L-BFGS optimizer, the F1 score follows a similar pattern to that of the previous data in which the score increases under an increase in number of hidden neurons and under a decrease of number of hidden layers. The SGD optimizer has the same trend for number of layers but no noticeable pattern for number of neurons. The computational runtime for L-BFGS increases proportionally with both axes and only with the number of layers for SGD. Like with the F1 score, there is no significant trend or pattern with the number of neurons in SGD.

For the SVM results in Fig. 3-12, validation error and F1 score once again share similar trends and values with one another. Despite containing significantly low values in these categories, the results under the RBF kernel bear similar patterns to the corresponding subfigures in the previous subsection, in which the metrics increase proportionally with both iterative parameters. Unlike before, however, these trends also apply to the linear kernel, although the corresponding curves are less smooth than their RBF counterparts. The runtime values, however, are drastically different from before. In this case, they more closely follow the accuracy and F1 plots in that runtime increases proportionally to each parameter. This once again varies between the two kernels in that RBF contains a smooth straightforward curve while linear is rougher and more oscillatory.

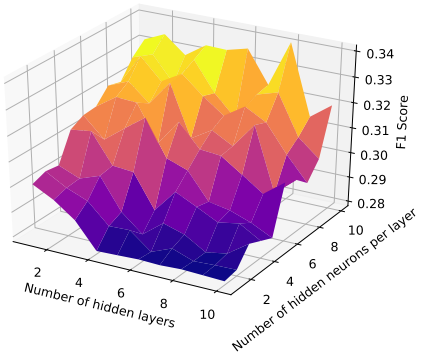
Fig. 3-13 depicts the KNN results, in which accuracy and F1 score follow an oscillatory pattern within a highly narrow range of values for nearly all parameter



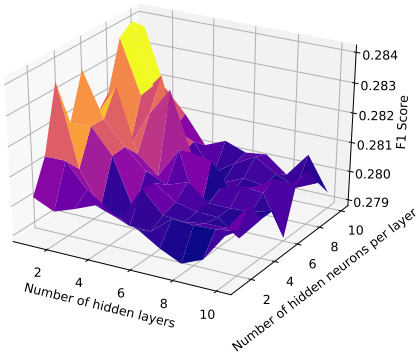
(a) Validation error for L-BFGS optimizer



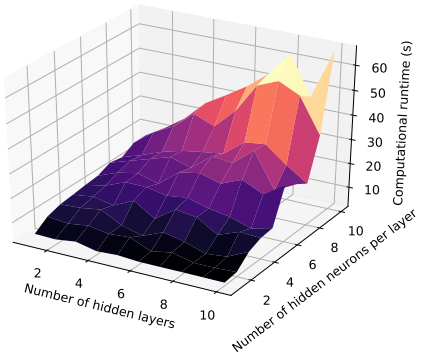
(b) Validation error for SGD optimizer



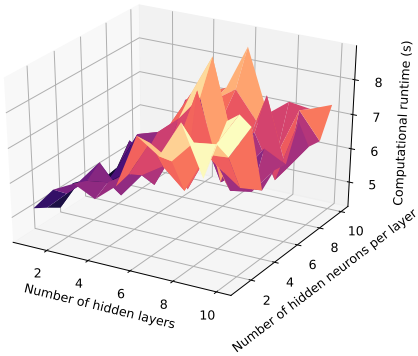
(c) F1 score for L-BFGS optimizer



(d) F1 score for SGD optimizer

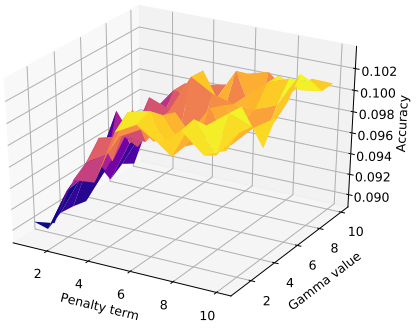


(e) Computational runtime for L-BFGS optimizer

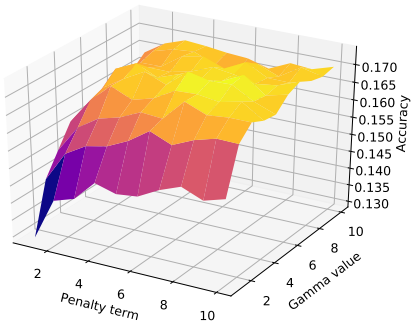


(f) Computational runtime for SGD optimizer

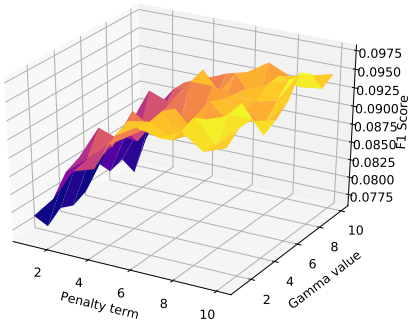
Figure 3-11: Plots of validation accuracy, F1 score, and computational runtime for ANN on DEAP data for L-BFGS and SGD optimizers.



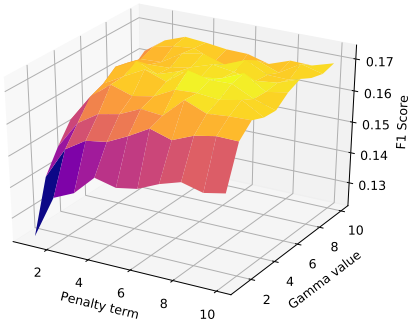
(a) Validation error for linear kernel



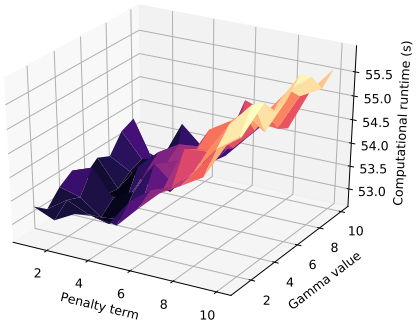
(b) Validation error for RBF kernel



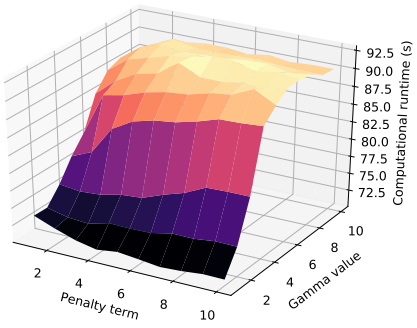
(c) F1 score for linear kernel



(d) F1 score for RBF kernel

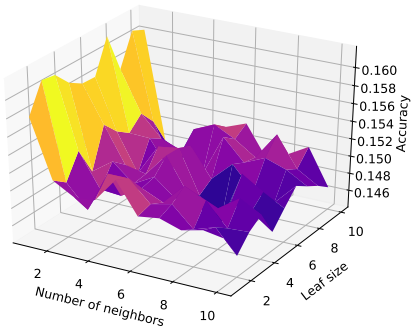


(e) Computational runtime for linear kernel

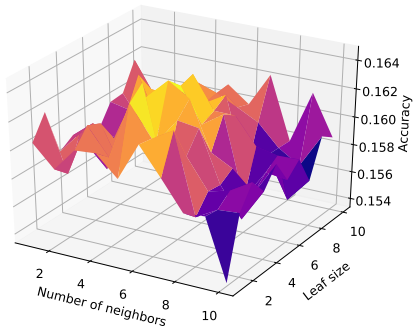


(f) Computational runtime for linear kernel

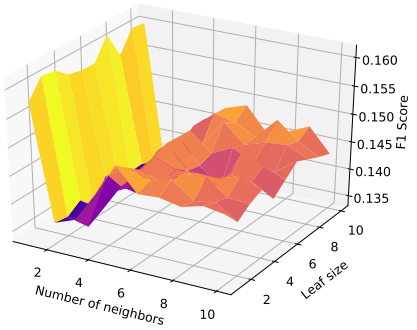
Figure 3-12: Plots of validation accuracy, F1 score, and computational runtime for SVM on DEAP data for linear and RBF kernels.



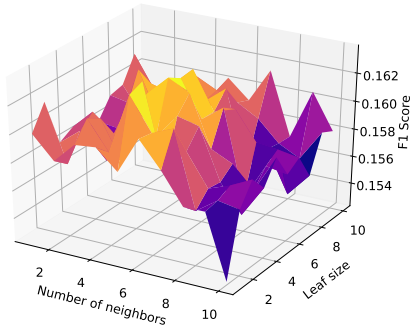
(a) Validation error for uniform weights



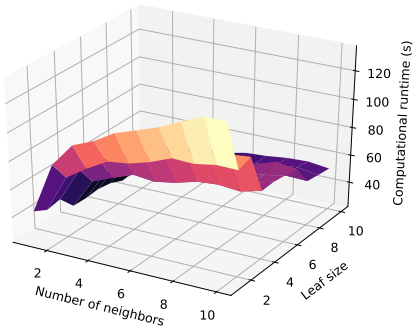
(b) Validation error for distance weights



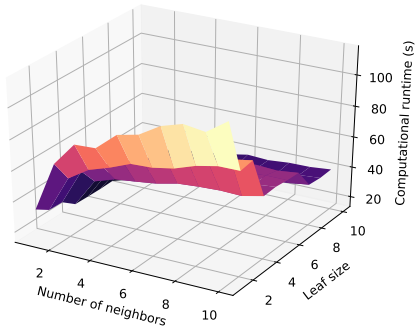
(c) F1 score for uniform weights



(d) F1 score for distance weights

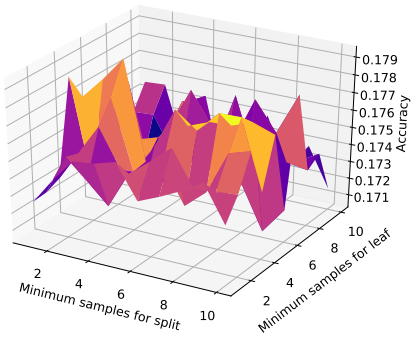


(e) Computational runtime for uniform weights

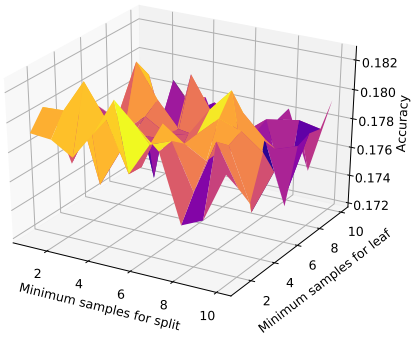


(f) Computational runtime for distance weights

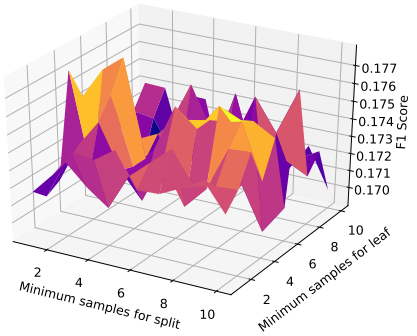
Figure 3-13: Plots of validation accuracy, F1 score, and computational runtime for KNN on DEAP data for uniform and distance weights.



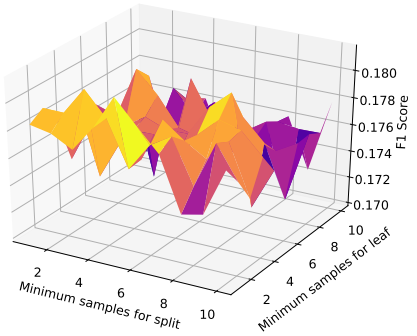
(a) Validation error for gini criterion



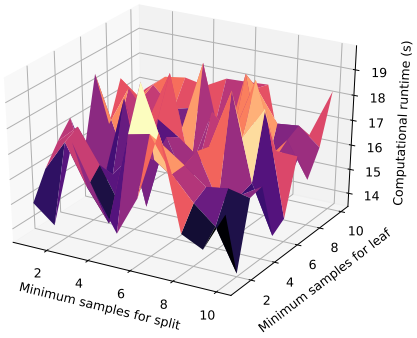
(b) Validation error for entropy criterion



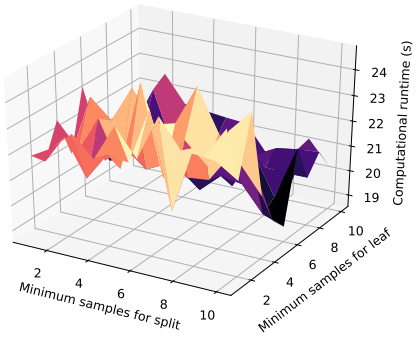
(c) F1 score for gini criterion



(d) F1 score for entropy criterion

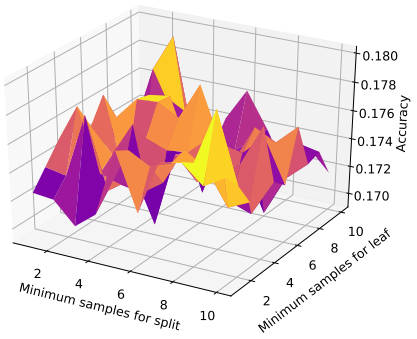


(e) Computational runtime for gini criterion

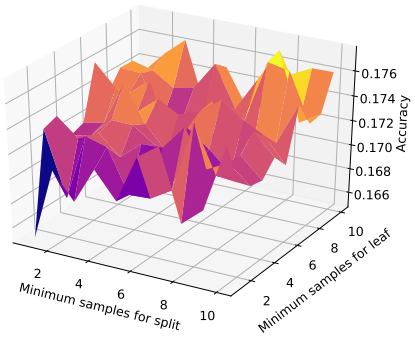


(f) Computational runtime for entropy criterion

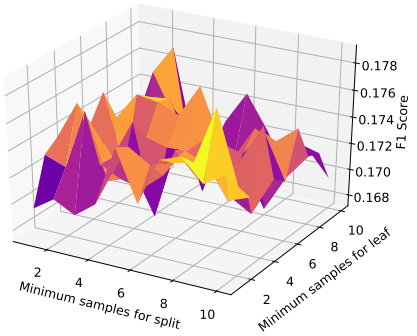
Figure 3-14: Plots of validation accuracy, F1 score, and computational runtime for DT on DEAP data for gini and entropy criteria.



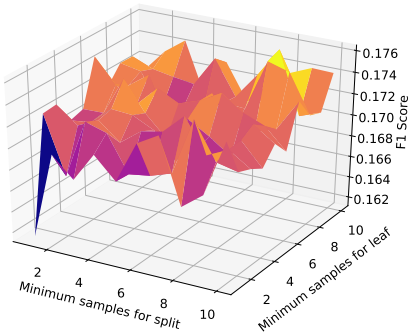
(a) Validation error for gini criterion



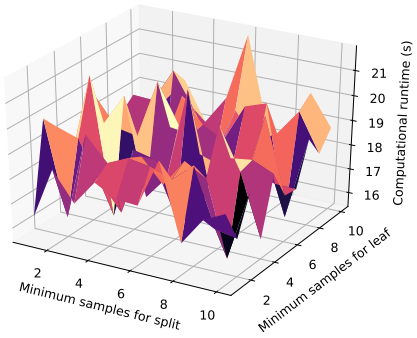
(b) Validation error for entropy criterion



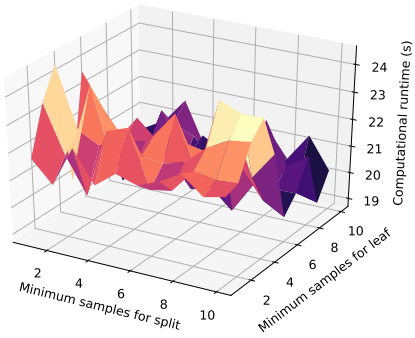
(c) F1 score for gini criterion



(d) F1 score for entropy criterion



(e) Computational runtime for gini criterion



(f) Computational runtime for entropy criterion

Figure 3-15: Plots of validation accuracy, F1 score, and computational runtime for RF on DEAP data for gini and entropy criteria.

Table 3.7: Summary of training accuracy, validation accuracy, and computational runtime on visual matching data.

	Train. Accuracy			Val. Accuracy			Runtime (s)		
Technique	Max	Min	Mean	Max	Min	Mean	Max	Min	Mean
ANN SGD	.726	.723	.724	.729	.718	.724	8.62	4.09	6.20
ANN L-BFGS	.738	.723	.729	.729	.717	.723	50.0	2.66	14.63
KNN Uniform	1.00	.749	.803	.721	.653	.702	167	24.9	77.7
KNN Distance	1.00	1.00	1.00	.716	.655	.693	161	23.4	74.5
SVM Linear	.725	.723	.724	.728	.720	.724	91.9	26.1	36.7
SVM RBF	.999	.735	.935	.738	.702	.723	181	60.8	138
DT Gini	1.00	.944	.960	.932	.889	.915	29.8	19.7	25.3
DT Entropy	1.00	.948	.968	.923	.896	.911	18.0	13.7	16.1
RF Gini	.979	.832	.899	.734	.718	.727	29.4	21.6	25.2
RF Entropy	.978	.843	.912	.733	.718	.727	18.9	13.5	16.2

combinations. The exception to this is under uniform weights when transitioning from one to two neighbors, in which there is a significant dip in accuracy. Such a significance, however, is merely relative to the range of values spanned by the remaining results, as the total difference in range is no more than .025, or 2.5%. Figs. 3-14 and 3-15 provide the equivalent results for DT and RF, respectively. Overall, there is nothing significant to observe about these plots besides the fact that are highly among a relatively short range of values for each metric. Given that each of the metrics has been unsuccessful in modeling this dataset, no further elaboration is needed.

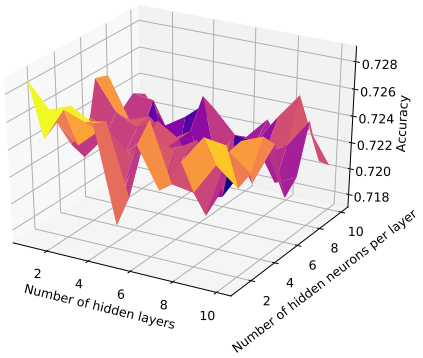
3.3.3 EEG Visual Matching Dataset

A summary of the accuracy and runtime values for the visual matching data is presented in Table 3.7. For the former category, results are significantly better than those of the DEAP dataset but not quite as high as in the arithmetic data. It is also worth noting that this is the first dataset in which the SVM linear kernel produces

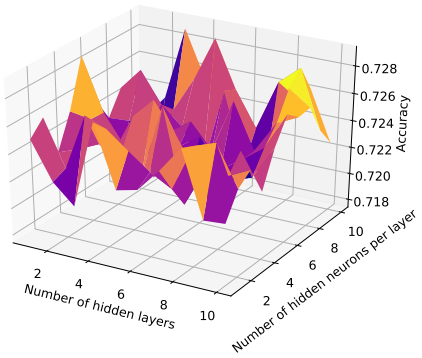
a minimally acceptable amount of accuracy, although in training it still produces the lowest range of accuracy values. In terms of training results, KNN under the distance weight configuration produces the best results possible, as every iterative parameter combination produces a perfect score. The next best training results stem from DT, as no validation value in this method falls below 90%. It is also the only method in which both subjective parameter options produces 90% training accuracy or better under all circumstances. Meanwhile, RF is the only other algorithm in which these results never drop below 80%. ANN and SVM linear kernel perform the worst, as training values are consistently in the range of 72-74%.

As for validation accuracy, DT is the only method with values above 74%. On top of that, all of its values are also above 88%, thereby easily making it the most ideal technique for accuracy alone in this dataset. For all the remaining methods, low validation accuracy was the result of either overfitting or similarly low training accuracy, depending on the parameters. Thus, conclusions can be reached that this data is easier to model than DEAP but more difficult than the arithmetic data. In terms of runtime, the best performance stemmed from ANN under the SGD optimizer. When coupled with mediocre accuracy, however, this trait is not particularly useful. The second best performer was a near tie between DT and RF. Given the high accuracy of DT, the former of the two methods is thereby observed to be the best balance so far of accuracy and runtime for this dataset.

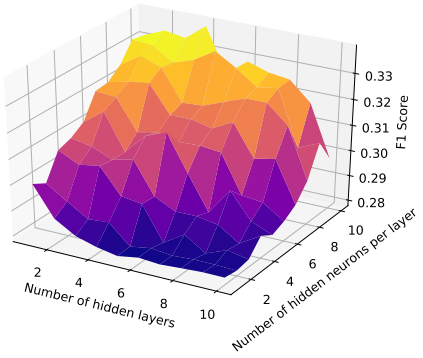
When observing similarly organized results for the other three metrics in Table 3.8, a stark difference is that none of the F1, precision, or recall results closely resemble the accuracy results, as was the case in many aspects of the previous two datasets. DT once again produces the highest scores across the board, but there is still a significant drop from accuracy values. This is likely due to the increased complexity of the data when compared to the arithmetic dataset. For all other methods, scores ranged anywhere from .240 to .660.



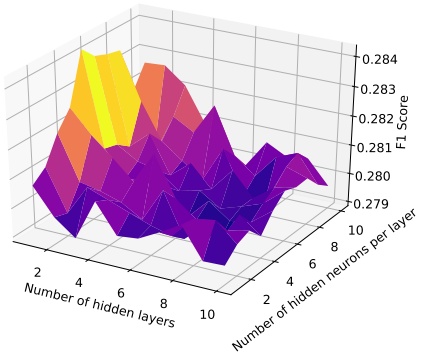
(a) Validation error for L-BFGS optimizer



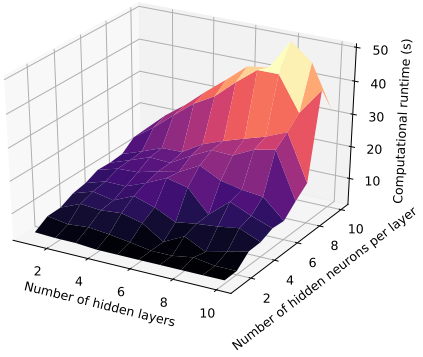
(b) Validation error for SGD optimizer



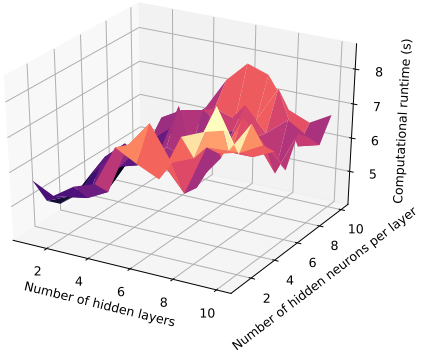
(c) F1 score for L-BFGS optimizer



(d) F1 score for SGD optimizer

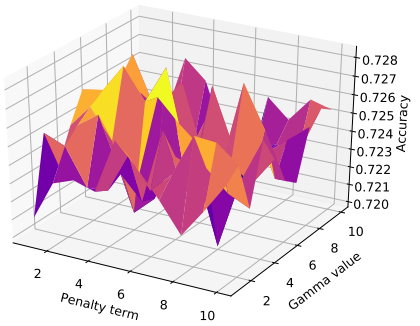


(e) Computational runtime for L-BFGS optimizer

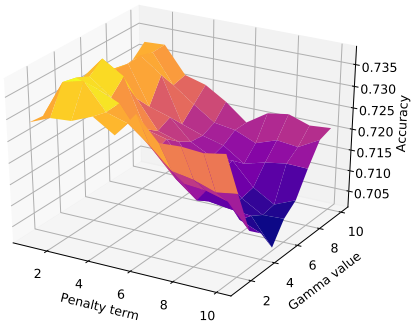


(f) Computational runtime for SGD optimizer

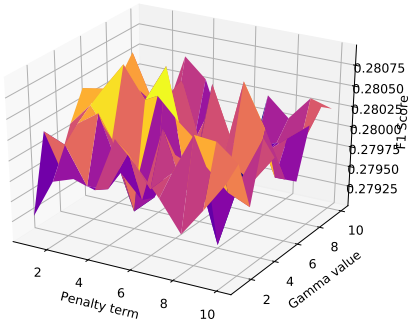
Figure 3-16: Plots of validation accuracy, F1 score, and computational runtime for ANN on visual matching data for L-BFGS and SGD optimizers.



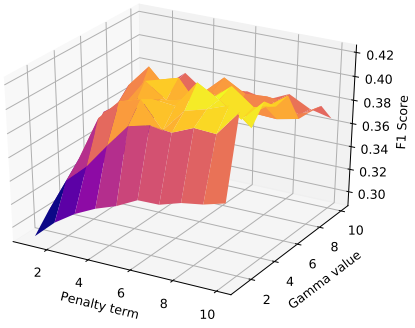
(a) Validation error for linear kernel



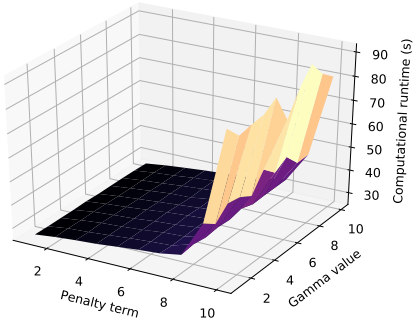
(b) Validation error for RBF kernel



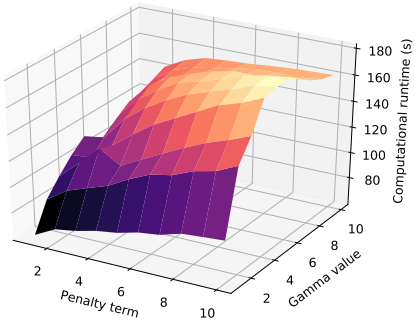
(c) F1 score for linear kernel



(d) F1 score for RBF kernel



(e) Computational runtime for linear kernel



(f) Computational runtime for linear kernel

Figure 3-17: Plots of validation accuracy, F1 score, and computational runtime for SVM on visual matching data for linear and RBF kernels.

Table 3.8: Summary of F1, precision, and recall scores on visual matching data.

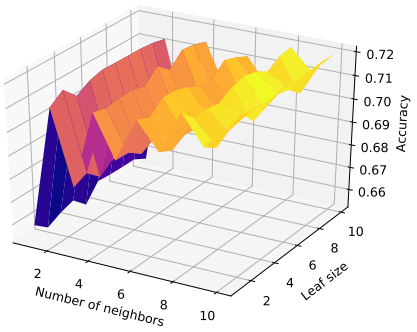
	F1			Precision			Recall		
Technique	Max	Min	Mean	Max	Min	Mean	Max	Min	Mean
ANN SGD	.284	.279	.280	.454	.240	.281	.335	.333	.334
ANN L-BFGS	.339	.279	.304	.454	.240	.347	.356	.333	.342
KNN Uniform	.419	.329	.359	.660	.379	.417	.417	.350	.366
KNN Distance	.417	353	.379	.640	.384	.417	.416	.361	.380
SVM Linear	.281	.279	.280	.243	.240	.241	.333	.333	.333
SVM RBF	.423	.290	.376	.643	.407	.497	.405	.337	.378
DT Gini	.631	.590	.605	.681	.594	.618	.621	.588	.600
DT Entropy	.619	.585	.602	.638	.585	.606	.621	.585	.600
RF Gini	.363	.335	.350	.438	.400	.421	.369	.356	.363
RF Entropy	.363	.335	.350	.438	.400	.421	.369	.356	.363

Figs. 3-16 and 3-17 contain this dataset's graphical results for ANN and SVM, respectively. For the ANN results, despite drastically different ranges of values, every pattern and trend observed in Fig. 3-11 holds true for every respective subfigure of Fig. 3-16. The only real change is that all the corresponding metric values are simply better. The SVM results, however, are drastically different from their DEAP counterparts. In the linear kernel, the validation error and F1 score plots closely resemble each other in that there is no noticeable pattern along the axes, just more oscillation among a narrow range of values. In the RBF kernel, the plots for the same two metrics are widely different from one another. Validation error appears to decrease as the penalty term increases, while the gamma value causes the function to decrease to a minimum when reaching a gamma value of five, then increases afterwards, essentially creating a valley area in the surface. The F1 score, however, follows a completely inverse trend, in which the score increases proportionally with the penalty term and a peak exists along the gamma value axis. As for runtime, the linear kernel contains a more unusual surface in which runtime is below 30 seconds and slowly increases

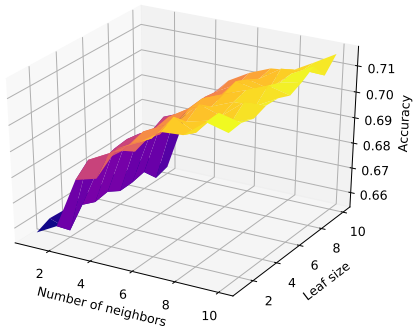
along the x axis until the penalty term reaches eight, at which point the slop increases drastically. Meanwhile, changes in gamma values seem to have to no effect on this metric. The linear kernel, however, produces a smooth surface with correlations closely resembling those of the F1 score but over a wider range of values, spanning from 60 to 190 seconds.

The KNN results in Fig. 3-18 present some more stark differences between validation accuracy and F1 score but similar trends between weight configurations. Under both metrics, however, the only common observation is that changes in the leaf size does not appear to impact the respective metrics. Accuracy appears to vary proportionally with the number of neighbors but with some oscillation. This oscillation is greater and more noticeable with uniform weights than it is with distance weights. The F1 score, however, varies inversely with the number of neighbors under both weight configurations. The runtime under both scenarios appears to be almost exactly the same under both uniform and distance weights, although there are in fact some slight numerical differences. In this metric, runtime decreases noticeably with leaf size and increases slightly with number of neighbors.

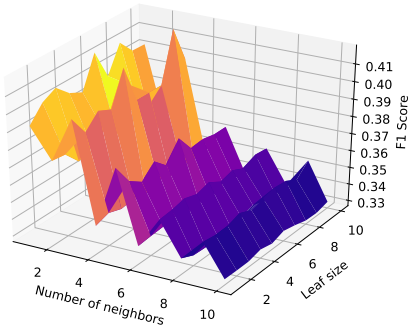
Finally, Figs. 3-19 and 3-20 depict the results for DT and RF, respectively. In both methods in each subjective parameter, validation error increases proportionally with the minimum number of samples for leaf, while the minimum number of samples for split have no noticeable correlation. The only notable difference between the plots for DT and RF is that the surfaces of the former are smoother and have less oscillation than the latter, particularly when compared to RF under the entropy criterion. The F1 score, however, varies substantially from method to method but not from criterion to criterion. In DT, there are no noticeable trends upon variation of the axes, whereas in RF, the score varies inversely with the minimum number of samples for leaf. Changes in the minimum samples for split, however, does not seem to affect the outcome. Meanwhile, computational runtime in all four cases has no real



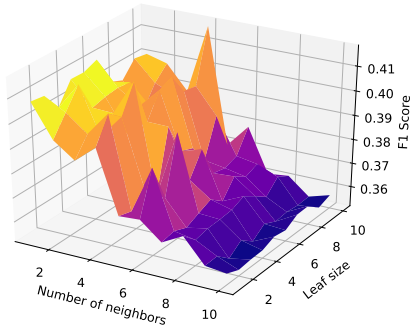
(a) Validation error for uniform weights



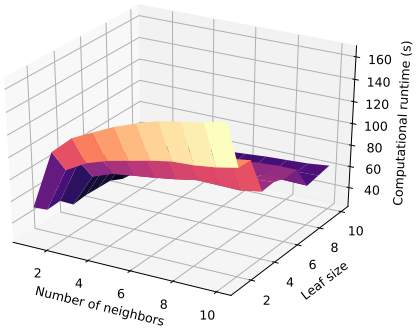
(b) Validation error for distance weights



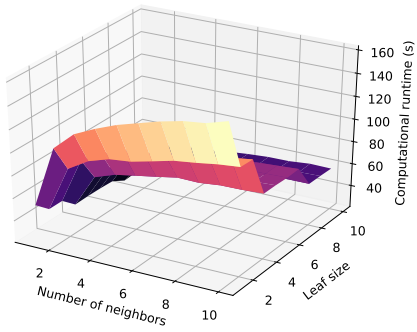
(c) F1 score for uniform weights



(d) F1 score for distance weights

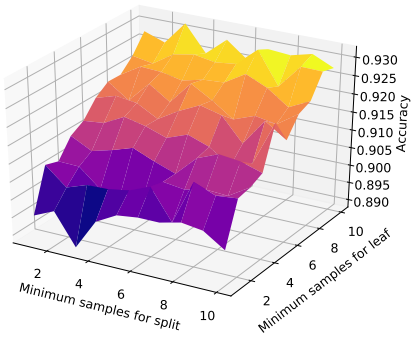


(e) Computational runtime for uniform weights

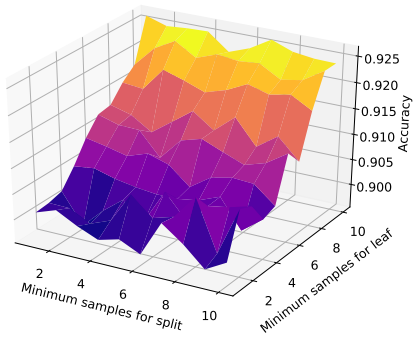


(f) Computational runtime for distance weights

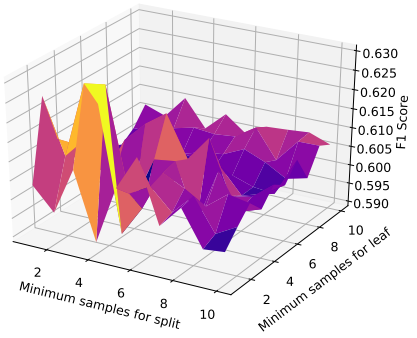
Figure 3-18: Plots of validation accuracy, F1 score, and computational runtime for KNN on visual matching data for uniform and distance weights.



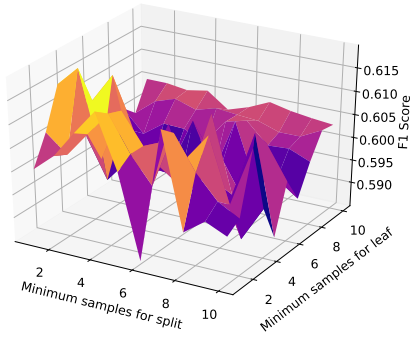
(a) Validation error for gini criterion



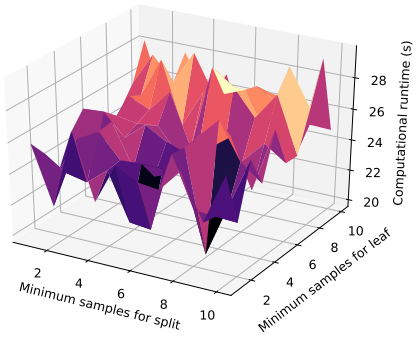
(b) Validation error for entropy criterion



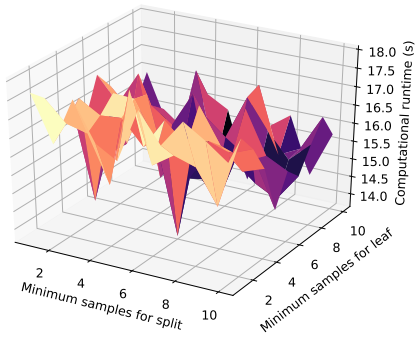
(c) F1 score for gini criterion



(d) F1 score for entropy criterion

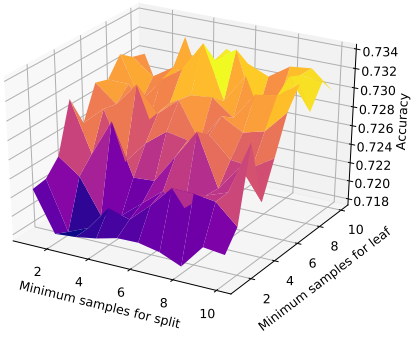


(e) Computational runtime for gini criterion

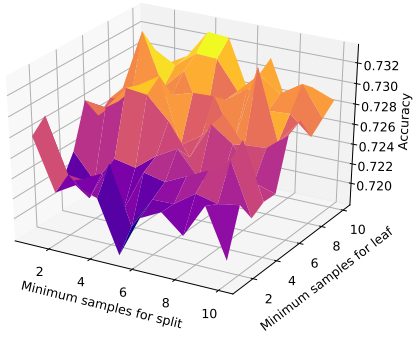


(f) Computational runtime for entropy criterion

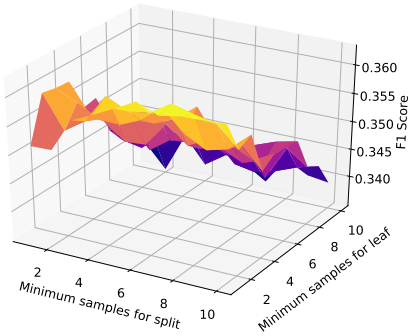
Figure 3-19: Plots of validation accuracy, F1 score, and computational runtime for DT on visual matching data for gini and entropy criteria.



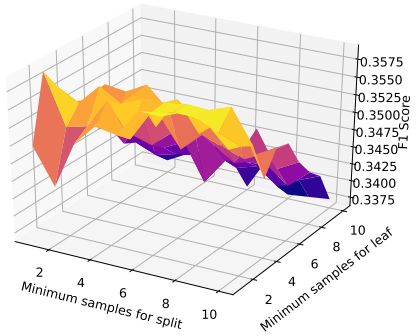
(a) Validation error for gini criterion



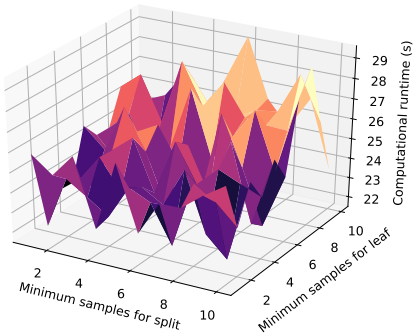
(b) Validation error for entropy criterion



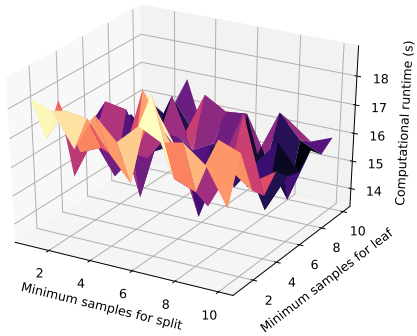
(c) F1 score for gini criterion



(d) F1 score for entropy criterion



(e) Computational runtime for gini criterion



(f) Computational runtime for entropy criterion

Figure 3-20: Plots of validation accuracy, F1 score, and computational runtime for RF on visual matching data for gini and entropy criteria.

Table 3.9: Summary of training accuracy, validation accuracy, and computational runtime on SAPHYRE data.

	Train. Accuracy			Val. Accuracy			Runtime (s)		
Technique	Max	Min	Mean	Max	Min	Mean	Max	Min	Mean
ANN SGD	.932	.080	.567	.930	.080	.565	66.2	9.69	28.0
ANN L-BFGS	.981	.081	.514	.973	.079	.511	137	3.88	69.3
KNN Uniform	1.00	.963	.980	.984	.949	.966	15.1	7.24	10.2
KNN Distance	1.00	1.00	1.00	.984	.970	.977	14.7	5.99	9.09
SVM Linear	.971	.948	.964	.967	.944	.960	12.2	8.30	9.47
SVM RBF	.999	.978	.995	.986	.972	.981	53.6	16.6	34.7
DT Gini	1.00	.991	.995	.993	.984	.989	10.7	6.62	8.34
DT Entropy	1.00	.991	.995	.993	.984	.989	9.07	6.27	7.59
RF Gini	1.00	.994	.997	.995	.987	.992	11.4	6.77	9.03
RF Entropy	1.00	.995	.998	.995	.989	.993	10.3	6.51	8.42

correlation with the x or y axis values, just oscillations among 4-10 second ranges of values.

3.3.4 SAPHYRE Pilot Dataset

Finally, the remainder of results left to observe involve the dataset of greatest importance to this chapter, the SAPHYRE dataset. This is because SAPHYRE is the only dataset to involve pilots, thereby containing the greatest relevance to the overarching dissertation topic of flight safety as well as to the current phase of the research based specifically on manned flight safety. Therefore, the successful ability to model this dataset under any of the ongoing experimental parameters is this chapter's highest priority.

Following the same pattern as in the previous three subsections, an overview of the runtime and accuracy results is given in Table 3.9. In terms of training accuracy, every method besides ANN produces consistently high accuracy, in that no training

accuracy value is below 94%. Another unique trend is that for three of the five methods, the maximum training accuracy reached is a perfect score. In addition, KNN under distance weights results in a perfect training score for every iterative parameter combination. In this data, ANN carries the distinction of having substantially high and substantially low accuracy values, ranging anywhere from 8% to 98%.

Turning to validation accuracy, the overall values for each method held up quite well. As the respective columns indicate, there is not a single trace of overfitting. Overall, RF produces the highest possible accuracy values as well as the highest minimum and average values. DT is a close second, followed by a near tie between KNN and SVM. ANN still produces a relatively high maximum value but still spans a wide range of results, this time varying from 8% to 93%.

In terms of computational runtime, ANN produces the maximum possible runtime as well as the minimum, continuing the trend so far of overwhelmingly wide metric ranges for ANN in this dataset. SVM has the second highest maximum value, while KNN has the second lowest minimum value. As for entire ranges of runtime results, DT produces the lowest, while SVM produces the highest. Overall so far, conclusions can be made that RF is the best method for accuracy, while DT is most ideal for speed and efficiency. In terms of optimal balance of speed and accuracy, DT has a slightly greater balance, although RF is a highly close second.

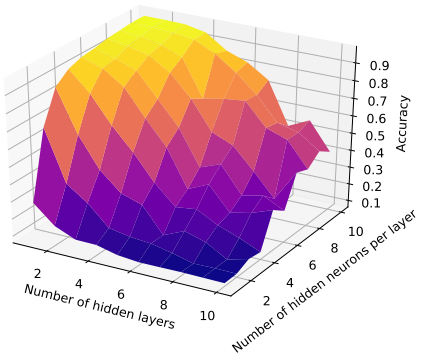
When observing the F1, precision, and runtime results in Table 3.10, ANN continues its tend of wide ranges of results, as its F1 score varies from 0.004 to 0.865, precision ranges from .002 to .880, and recall varies slightly less drastically from .027 to .865. In all three metrics, RF excels at all three with the best maximum, minimum, and average values for each category. SVM under the linear kernel has the consistently worst values, while DT has the second best. Thus, when identifying the most ideal method for the SAPHYRE data under all six metrics, RF has the best balance of all six result categories, while DT is the slightly balance of accuracy and

Table 3.10: Summary of F1, precision, and recall scores on SAPHYRE data.

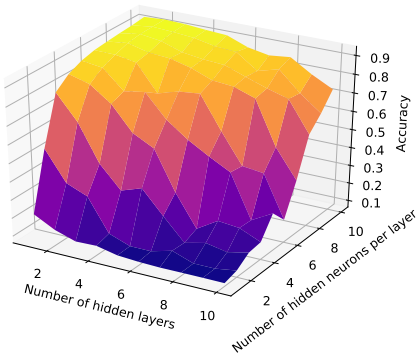
	F1			Precision			Recall		
Technique	Max	Min	Mean	Max	Min	Mean	Max	Min	Mean
ANN SGD	.865	.004	.429	.880	.002	.437	.865	.027	.451
ANN L-BFGS	.962	.004	.376	.964	.002	.378	.961	.027	.401
KNN Uniform	.981	.940	.959	.981	.947	.963	.981	.935	.956
KNN Distance	.981	.965	.973	.982	.967	.975	.981	.963	.972
SVM Linear	.953	.906	.940	.959	.930	.950	.950	.904	.937
SVM RBF	.984	.968	.977	.985	.971	.982	.983	.960	.973
DT Gini	.991	.976	.982	.992	.976	.984	.991	.976	.982
DT Entropy	.991	.975	.984	.991	.976	.985	.991	.975	.984
RF Gini	.992	.982	.988	.994	.986	.990	.991	.979	.986
RF Entropy	.993	.983	.989	.994	.987	.991	.992	.979	.987

runtime alone.

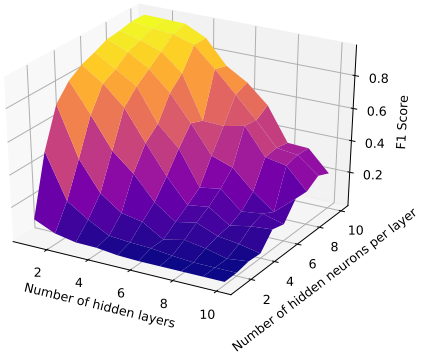
Shifting once more to graphical results, the vast ranges of metric values obtained by ANN on the SAPHYRE data can be observed more clearly in Fig. 3-21. This is evident in all three visible metrics, with accuracy and F1 ranges from 0.1 to 0.9 and with runtime from three seconds to two minutes. In terms of variability among axes, all six subfigures follow the same trends as their arithmetic data counterparts but over a wider field of values. Meanwhile, the SVM results in Fig. 3-22 bear some resemblance to those of the DEAP dataset. When evaluating accuracy and F1 score with the linear kernel, the same trends hold as with their DEAP counterparts. In the RBF kernel, similar trends emerge, except that the gamma value axis results in a rise and then fall of the respective metric values, essentially creating a peak halfway through the axis. In terms of runtime, the RBF kernel continues the same trends as with DEAP, but the linear kernel follows an entirely new correlation. In this case, the runtime decreases as the penalty term increases, while the change in gamma value has little to no effect on the metric result.



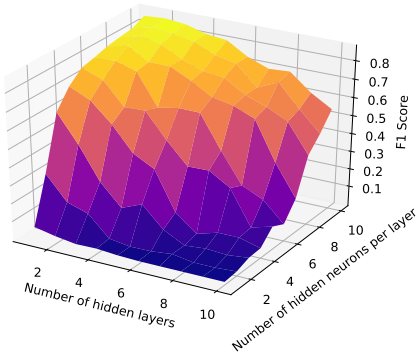
(a) Validation error for L-BFGS optimizer



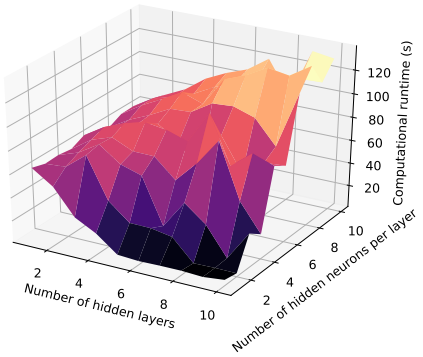
(b) Validation error for SGD optimizer



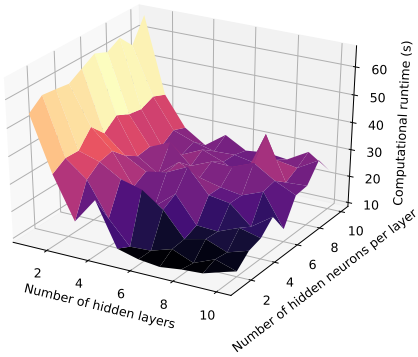
(c) F1 score for L-BFGS optimizer



(d) F1 score for SGD optimizer

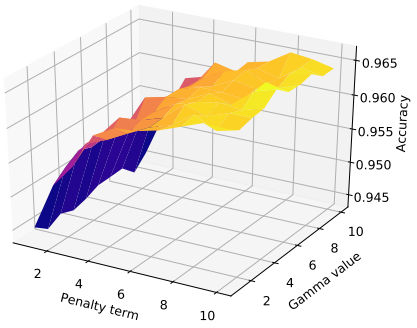


(e) Computational runtime for L-BFGS optimizer

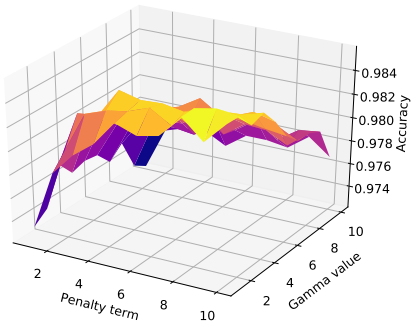


(f) Computational runtime for SGD optimizer

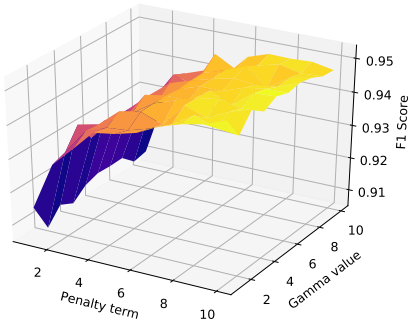
Figure 3-21: Plots of validation accuracy, F1 score, and computational runtime for ANN on SAPHYRE data for L-BFGS and SGD optimizers.



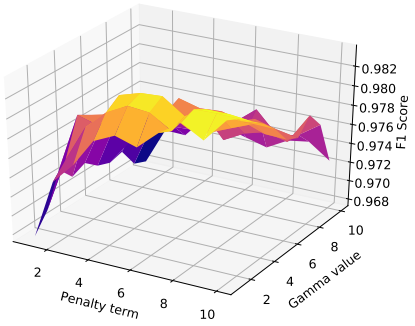
(a) Validation error for linear kernel



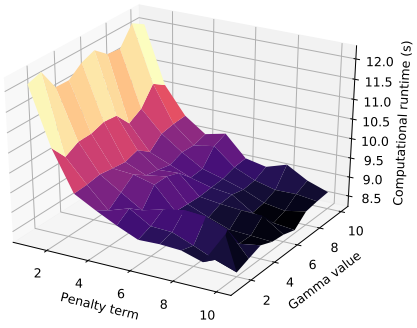
(b) Validation error for RBF kernel



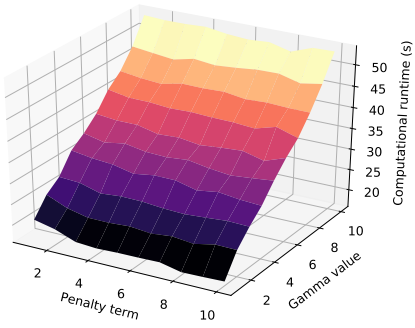
(c) F1 score for linear kernel



(d) F1 score for RBF kernel

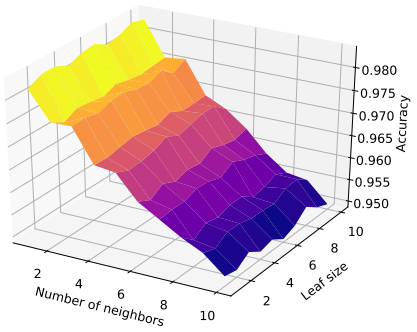


(e) Computational runtime for linear kernel

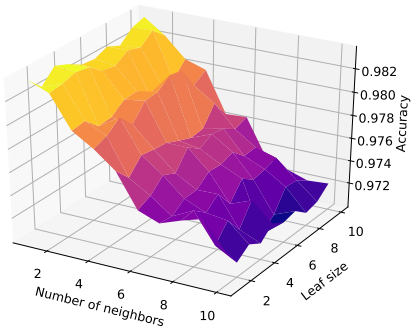


(f) Computational runtime for linear kernel

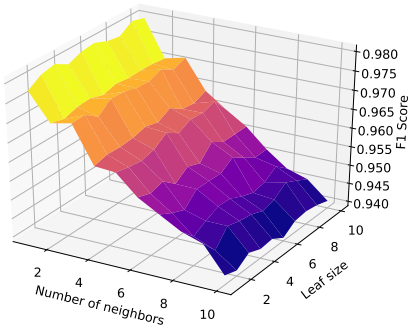
Figure 3-22: Plots of validation accuracy, F1 score, and computational runtime for SVM on SAPHYRE data for linear and RBF kernels.



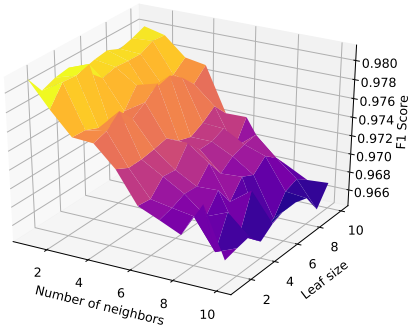
(a) Validation error for uniform weights



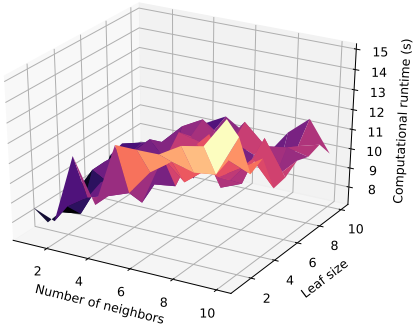
(b) Validation error for distance weights



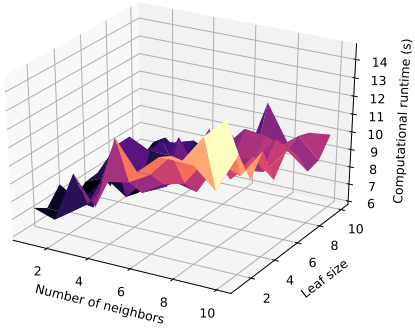
(c) F1 score for uniform weights



(d) F1 score for distance weights



(e) Computational runtime for uniform weights



(f) Computational runtime for distance weights

Figure 3-23: Plots of validation accuracy, F1 score, and computational runtime for KNN on SAPHYRE data for uniform and distance weights.

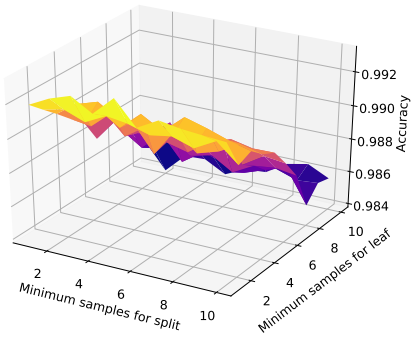
The plots of the KNN results are given in Fig. 3-23. As the first four subfigures indicate, accuracy and F1 score are influenced only by the number of neighbors by an inverse proportionality and little to none by leaf size. The runtime results follow the reverse trend from the perspective of the former axis in that runtime increases with the number of neighbors. All of these observations hold true under both weight configuration options.

Lastly, the DT and RF results are displayed in Figs. 3-24 and 3-25, respectively. Both figures follow similar trends in each respective subfigure in that an increased minimum number of samples for leaf result in decreased accuracy and F1 score, while changes in the minimum number of samples for split do not have any noticeable effect. Runtime, however, does not follow any meaningful correlations with either axis and simply oscillates among up to four-second ranges of runtime values.

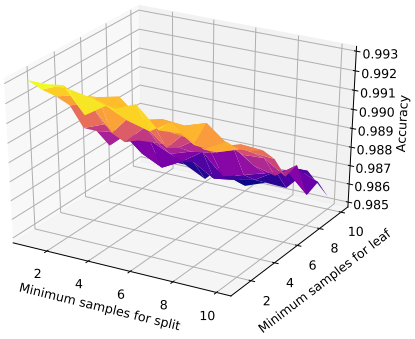
3.4 Discussion

Given the sheer volume of results and parameters in the previous section, it is useful to begin analysis and discussion in a tabular format. Thus, five tables are presented that summarize the advantages and disadvantages of the different ML techniques on the four datasets. The first four tables are specific to each dataset, while the last one is a full comprehensive comparison among all the data.

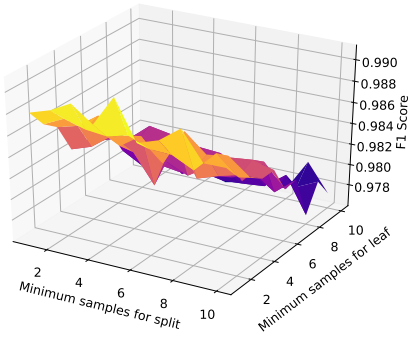
Table 3.11 presents an overview of the technique comparison on the arithmetic dataset. As mentioned previously, DT provides the best results in nearly every metric. Within this method, gini provides a slightly better balance of accuracy and runtime. Other observations include the fact that ANN has a wide range of results and that SVM with the linear kernel provides substantially poor results when compared to any other compared for this data. Similar results for the DEAP dataset are given in Table 3.12. In this case, no method produced substantially high results, but ANN



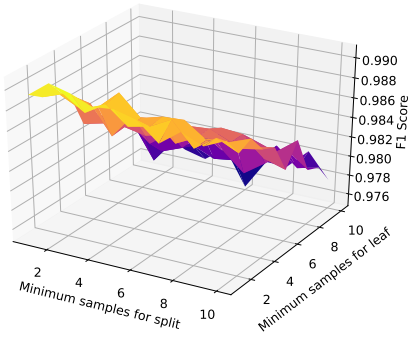
(a) Validation error for gini criterion



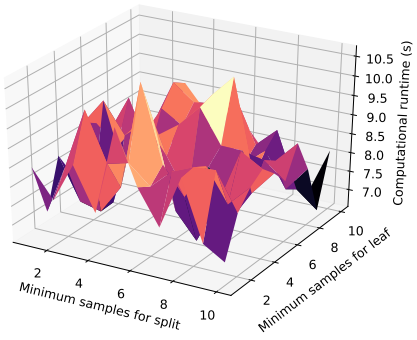
(b) Validation error for entropy criterion



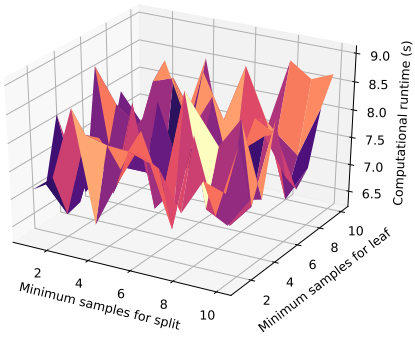
(c) F1 score for gini criterion



(d) F1 score for entropy criterion

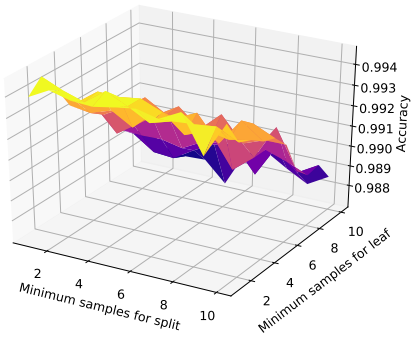


(e) Computational runtime for gini criterion

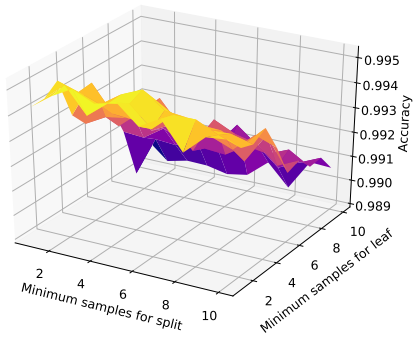


(f) Computational runtime for entropy criterion

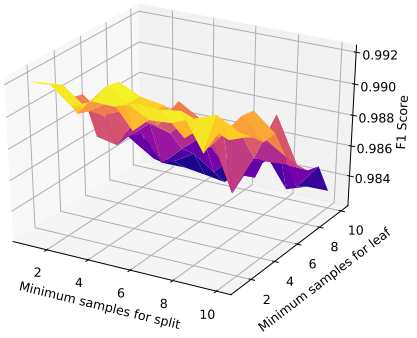
Figure 3-24: Plots of validation accuracy, F1 score, and computational runtime for DT on SAPHYRE data for gini and entropy criteria.



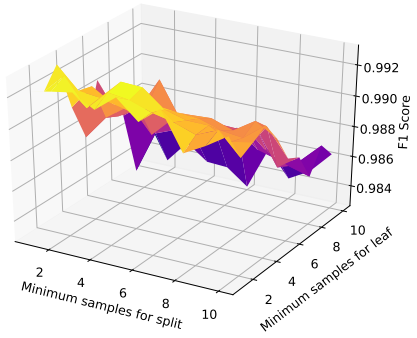
(a) Validation error for gini criterion



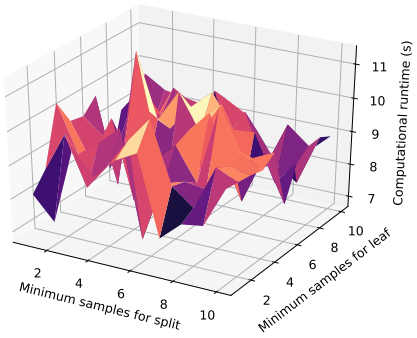
(b) Validation error for entropy criterion



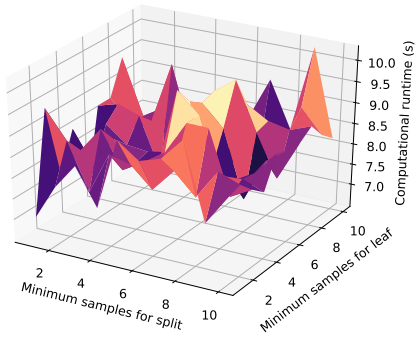
(c) F1 score for gini criterion



(d) F1 score for entropy criterion



(e) Computational runtime for gini criterion



(f) Computational runtime for entropy criterion

Figure 3-25: Plots of validation accuracy, F1 score, and computational runtime for RF on SAPHYRE data for gini and entropy criteria.

Table 3.11: Summary of technique comparison on arithmetic data.

Technique	Advantages	Disadvantages
ANN SGD	High accuracy at NH = 10 and NL = 1	Wide variation in accuracy
ANN L-BFGS	High accuracy/low runtime at same points	Wider variation in accuracy
KNN Uniform	Stable accuracy at most parameters	Third lowest max. F1 score
KNN Distance	Perfect training accuracy scores	Third lowest max. precision/recall
SVM Linear	Little variation in metric values	Lowest non-runtime values in every category
SVM RBF	Better results than linear kernel	Third highest max. runtime
DT Gini	Best values overall	Nothing noteworthy
DT Entropy	Best values except runtime	Slightly higher runtime than gini
RF Gini	Second highest accuracy values	Not as good as DT
RF Entropy	Third highest accuracy values	Not as good as DT

Table 3.12: Summary of technique comparison on DEAP data.

Technique	Advantages	Disadvantages
ANN SGD	Best accuracy and runtime	Results still subpar
ANN L-BFGS	Best max. values in most other metrics	High max. runtime
KNN Uniform	Highest max. training accuracy	Highest max. runtime, overfitting
KNN Distance	Perfect training accuracy	Overfitting
SVM Linear	Not the worst runtime results	Consistently poor results
SVM RBF	High max. training accuracy	Overfitting
DT Gini	Highest max. training accuracy	Overfitting
DT Entropy	Highest max. training accuracy	Overfitting
RF Gini	Highest max. training accuracy	Overfitting
RF Entropy	Highest max. training accuracy	Overfitting

gives the only minimal acceptable accuracy values. Linear SVM still provides similarly poor results, while all other methods fall victim to substantial amounts of overfitting, thereby making ANN the only useful method for this dataset.

Results for the visual matching data are presented next in Table 3.13. Like with the arithmetic data, DT provides the best results in nearly all metrics, while ANN is best for runtime only. Linear SVM does better here than in the past two datsets but still not substantially well. Meanwhile, results for the SAPHYRE data are given in Table 3.14. Here, there are fewer significant observations to make, as many methods

Table 3.13: Summary of technique comparison on visual matching data.

Technique	Advantages	Disadvantages
ANN SGD	Lowest runtime value range	Poor F1, precision, recall
ANN L-BFGS	Lowest minimum runtime	Poor F1, precision, recall
KNN Uniform	Highest max. training accuracy	Second highest runtime range
KNN Distance	Perfect training accuracy	Third highest runtime range
SVM Linear	Better results than in last two datasets	Lowest F1, precision, recall
SVM RBF	Second highest max. training accuracy	Highest runtime range
DT Gini	Highest in most metrics	Results still worse than in previous dataset
DT Entropy	Lowest runtime	Results still worse than in previous dataset
RF Gini	Third lowest max. runtime	Poor F1, precision, recall
RF Entropy	Second lowest runtime	Poor F1, precision, recall

have nothing noteworthy to report. ANN provides the widest range of results, while RF provides the best results overall.

Finally, Table 3.15 gives a complete overview of method comparisons across all four datsets. From here, some broad and general observations can be made. For one, DT and RF seem to have the best mix of results in most cases except on the DEAP data, in which case ANN does best. SVM tends the fare on the low end of most metrics, including runtime of the RBF kernel and most other metrics under the linear kernel. ANN typically produces a wide variety of results ranging from substantially high values to substantially low ones.

3.5 Conclusion

This chapter presented a comprehensive analysis of the advantages and disadvantages of five candidate machine learning techniques on cognitive workload assessment. This was conducted on four relevant datasets under a variety of different experimental parameters. The results showed decision trees as the best overall technique for the arithmetic and visual matching datasets, random forest for SAPHYRE data, and artificial neural networks for the DEAP data.

Table 3.14: Summary of technique comparison on SAPHYRE data.

Technique	Advantages	Disadvantages
ANN SGD	Minimal loss from train. to val.	Lowest max. F1, precision, recall
ANN L-BFGS	Lowest min. runtime	Highest max. runtime
KNN Uniform	Highest max. train. accuracy	Nothing noteworthy
KNN Distance	Perfect train. accuracy	Nothing noteworthy
SVM Linear	Nothing noteworthy	Nothing noteworthy
SVM RBF	Nothing noteworthy	Second highest average runtime
DT Gini	Highest max. train. accuracy	Nothing noteworthy
DT Entropy	Lowest runtime	Nothing noteworthy
RF Gini	Highest max. train. accuracy	Nothing noteworthy
RF Entropy	Highest val. accuracy	Nothing noteworthy

Table 3.15: Summary of technique comparison on all data.

Technique	Arithmetic	DEAP	Visual Matching	SAPHYRE
ANN SGD	Wide range of results	Best in most metrics	Best runtime	Widest range of results
ANN L-BFGS	Wide range of results	Best in most metrics	Nothing noteworthy	Highest max. runtime
KNN Uniform	Most susceptible to overfitting	Worst max. runtime	Overfitting	Nothing noteworthy
KNN Distance	Most susceptible to overfitting	Second worst max. runtime	Overfitting	Nothing noteworthy
SVM Linear	Worst in all metrics	Worst in most metrics	Nothing noteworthy	Nothing noteworthy
SVM RBF	Worst maximum runtime	Worst runtime	Worst runtime	Nothing noteworthy
DT Gini	Best in all metrics	Nothing noteworthy	Best in most metrics	Nothing noteworthy
DT Entropy	Best in all except runtime	Nothing noteworthy	Second best in most metrics	Best runtime
RF Gini	Second best in most metrics	Nothing noteworthy	Overfitting	Best max. accuracy
RF Entropy	Third best in most metrics	Nothing noteworthy	Overfitting	Best in most metrics

From here, the remainder of the experiment is to apply these observations relative to their datasets as well as the specific attributes of the datasets themselves to conclude which aspects of the various datasets result in the different optimal choices of machine learning techniques. For that, it is helpful to review how each dataset differs from one another. The arithmetic data has the fewest inputs of all four and five possible output classifications. Thus, most accuracy results were consistently high, and computational runtime was relatively short. In this case, decision trees produced the best results with random forest as a close second. DEAP, meanwhile, was the most difficult data to model, containing 42 inputs and 40 possible output classifications.

In this case, no method was stellar, but ANN produced the best results overall. The visual matching data, however, has even more inputs at a total of 66. However, with only three possible classifications, this data was somewhat easier to model than the previous set, with DT again producing most of the best results. Thus, it achieved some reasonably high accuracy values, but some of the other metrics, including precision, runtime, and F1 score, were substantially lower. Finally, the SAPHYRE data had the second lowest amount of inputs at 16, but it also has the highest amount of possible output classifications, a grand total of 165. In this case, RF produces the best results.

These observations will provide a significant first step in the overall scope of this research, as the different machine learning techniques can be analyzed and selected based on the cognitive workload data being given. By being able to dynamically decide the most accurate and most efficient technique at any given point in time, early detection of mental overload and underload given some physiological data can be done easily and quickly in order to most effectively optimize the human factors aspects of manned flight safety.

Chapter 4

Human-Side Flight Performance through Adaptive Workload Prediction

The previous chapter conducted a thorough evaluation of different machine learning mechanisms in the application of cognitive workload. Following such an analysis, the next logical step in the context of this research is to utilize these advantages and disadvantages to generate a system that dynamically selects a technique for prediction of cognitive workload based on the attributes of a given dataset. From there, the proposed algorithm can provide early detection and warning of cognitive overload or underload. Once this warning takes place, a pilot can attempt to correct his or her mental load balance through sliding scale autonomy adjustment or other methods. Thus, this chapter presents the theory, process, results, and discussion of an adaptive machine learning algorithm for dynamic early detection of cognitive load.

Table 4.1: Summary of cognitive workload dataset results.

Dataset	No. Inputs	No. Classifications	Best Method	Optimal Parameters
Arithmetic	3	5	DT	gini, 9 for split, 2 for leaf
Visual Matching	66	3	DT	gini, 9 for split, 2 for leaf
DEAP	42	40	ANN	SGD, 7 neurons, 2 layers
SAPHYRE	16	165	RF	entropy, 7 for split, 10 for leaf

4.1 Methods and Materials

The experimental setup of this chapter bears much similarity to that of the previous chapter. For instance, the proposed algorithm was programmed in Python, again using the scikit-learn toolkit for quick access to different machine learning tools under the same hardware and software environments described in Section 3.2.

In formulating the algorithm, there are two initial approaches that can be followed. One is to pick an ML technique based on the attributes of the dataset and run the one selection throughout every data sample. Another is to train multiple possible ML classifiers at the initial stage of the algorithm and dynamically apply different techniques to different points in time based on which method is expected to produce the best accuracy. The advantage of the first approach is the greater amount of computational efficiency involved in only needing to train one model, but this is paired with the drawback of having a level of accuracy limited to the capabilities of the one selected method. The second approach remedies this issue by having the ability to dynamically switch between ML methods when accuracy of the chosen technique falls behind that of another. The drawback of this, however, lies within the computational complexity involved in training multiple different ML models for the same dataset and evaluating multiple models simultaneously at a single time step. For comparative purposes, this chapter will explore both approaches.

For both cases, the fundamental initial step is to set the conditions in which one ML technique is preferred over the others. To do so, it is useful to revisit the

Table 4.2: Simplified summary of cognitive workload dataset results.

Dataset	No. Inputs	No. Classifications	Best Method
Arithmetic	Lo	Lo	DT
Visual Matching	Hi	Lo	DT
DEAP	Hi	Hi	ANN
SAPHYRE	Lo	Hi	RF

conclusions of the previous chapter. Table 4.1 provides a summary of the four datasets as well as most optimal machine learning technique recommended for each one. When observing the numerical characteristics of each set of data, it useful to rewrite and reorganize the results into a simpler form as portrayed in Table 4.2. As the results in this table indicate, there is a coincidentally balanced combination of attributes in each dataset, in which number of inputs and number of possible output classifications is either large large or small. Out of the four datasets, each combination of attributes has either two large values, two small values, or a combination of each. Hence, this approach is highly beneficial in determining the thresholds in which to select a particular ML method. In this case, 20 appears to be a well-rounded threshold value that can effectively separate high values from low values. Recall from the end of Chapter 3 that SVM and KNN did not produce any substantially satisfactory results. Hence, these two methods will not be applied here. With the threshold criteria intact, the next step is the full development of the two algorithmic approaches.

The single-technique approach can be presented quite simply as given in Fig. 4-1. The more adaptive approach, however, requires training of all three ML techniques as well as a sample-by-sample analysis of the data to dynamically allocate the best method for the best points in time. This more elaborate algorithm is presented in Fig. 4-2. From there, the final step is to design the iterative experimental process for both approaches on all four datasets under any additional parameters that may be deemed useful in evaluating the effectiveness of these approaches. The complete

```
train_in, val_in, train_out, val_out ← all possible input and output data
if number of output classifications < 20 then
   create DT classifier
else
   if number of inputs > 20 then
      create ANN classifier
   else
      create RF classifier
   end if
end if
train selected classifier
for all validation time steps do
   get validation accuracy for all time steps evaluated thus far
end for
return  val_accuracy, method, runtime, train_accuracy
```

Figure 4-1: Pseudocode of single selection process.

```
train_in, val_in, train_out, val_out ← all possible input and output data
create and train ANN, DT, and RF classifiers
if number of output classifications < 20 then
   select DT
else
   if number of inputs > 20 then
      select ANN
   else
      select RF
   end if
end if
for all validation time steps do
   get validation accuracy from selected method for all completed time steps
   if accuracy is lower than that of another method then
      select method with highest accuracy
   end if
end for
return  val_accuracy, method, runtime, train_accuracy
```

Figure 4-2: Pseudocode of dynamic selection process.

```
1: for all datasets do
2:   for all methods do
3:     for all validation ratios do
4:       for all trials do
5:         results_s ← single_selection_process
6:         results_d ← dynamic_selection_process
7:       end for
8:       for all results do
9:         results_s ← results_s / no_iterations
10:        results_d ← results_d / no_iterations
11:        save results_s and results_d
12:      end for
13:    end for
14:  end for
15: end for
```

Figure 4-3: Pseudocode of complete experimental process.

experimental process is depicted in Fig. 4-3. In this case, the only metrics of relevance are training accuracy, validation accuracy, and runtime. In addition, different ratios of training data to validation data are also evaluated. Initial focus is on a more typical ratio of 25% validation followed by a more restrictive ratios of 66% validation, an amounts that is normally uncommon in machine learning applications. This is to provide further analysis on the effectiveness of the proposed algorithms and to test the limits of algorithmic stability. In the case of this chapter, the desired inputs and outputs for these algorithms are generic due to the vastly differing variable types in the different datasets. The output is still a single numerical classification, while the number of inputs can range from 3 to 66.

4.2 Experimental Results

Throughout the experimental results, the two algorithms will be referred as: 1) the single selection algorithm, as one ML method is selected for the entire duration of the process, and 2) the dynamic selection algorithm, as the ML technique being

Table 4.3: Summary of training accuracy, validation accuracy, and computational runtime under the single selection algorithm with 25% of samples used for validation.

Metric	Arithmetic	Visual Matching	SAPHYRE
Training Accuracy	.999	.967	.995
Validation Accuracy	.999	.902	.989
Runtime (s)	12.1	36.5	129

utilized for classification is subject to change at any given point in time. From here, the remainder of the experimental results is divided into two subsections, one for each validation ratio.

4.2.1 25% Validation

The results for the single selection algorithm at 25% validation are presented in Table 4.3. The most prominent observation upon initial analysis of results is that the DEAP data consistently produced extremely low training and validation accuracy in the magnitude of below 5%. It is worth keeping in mind that in the previous chapter, ANN was the only machine learning technique that produced more than 70% validation accuracy, while all other methods produced accuracy below 20%. Given DEAP's performance, here it can be concluded that the dataset under any of the parameters given in this chapter or the last is simply too unreliable to be properly evaluated in this algorithm. Hence the remainder of the chapter will only focus on the remaining three datasets.

In a manner predictable to the results in the previous chapter, the arithmetic dataset contains the highest training and validation accuracy of the three with values consistent with the maximum values obtained in Table 3.3. In addition, the runtime is the lowest due to the data having the least amount of inputs. The SAPHYRE data contains the next highest accuracy. The validation accuracy in this case is within

Table 4.4: Summary of training accuracy, validation accuracy, and computational runtime under the dynamic selection algorithm with 25% of samples used for validation.

Metric	Arithmetic	Visual Matching	SAPHYRE
RF Train. Accuracy	.998	.845	.995
DT Train. Accuracy	.999	.974	.997
ANN Train. Accuracy	.444	.724	.161
Validation Accuracy	.999	.902	.991
Runtime (s)	134	147	241

the range of best validation accuracy values generated in Table 3.9. The training accuracy, while not in the maximum training range given in said table (which in such case would be all ones) is well within the range of training values adjacent to the optimal validation accuracy values. The lowest accuracy value is from the visual matching data, although it still above 90% and also within the maximum validation accuracy range as given in Table 3.7. This dataset also outperforms the SAPHYRE data in runtime despite having more inputs, likely due to the smaller number of possible output classifications. Overall, these results indicate that the single selection algorithm is highly effective at bringing high accuracy to each of the datsets.

From there, Table 4.4 provides similar results for the dynamic selection algorithm. In terms of organization of results, the key difference here is that the training results are provided for all three candidate ML methods. For the arithmetic data, both RF and DT provide high training accuracy, although DT has a slight edge and is chosen due to the dataset's attributes. In this case, the validation accuracy matches that in Table 4.3. Given the substantial increase in runtime, it can be inferred that the single selection technique is more ideal for this particular data. The visual matching data also holds this trend, containing just over 90% validation accuracy under both algorithms with a significant difference in runtime. The SAPHYRE data, however, contains a slight increase in accuracy at a lower increase of runtime. Thus, this dataset

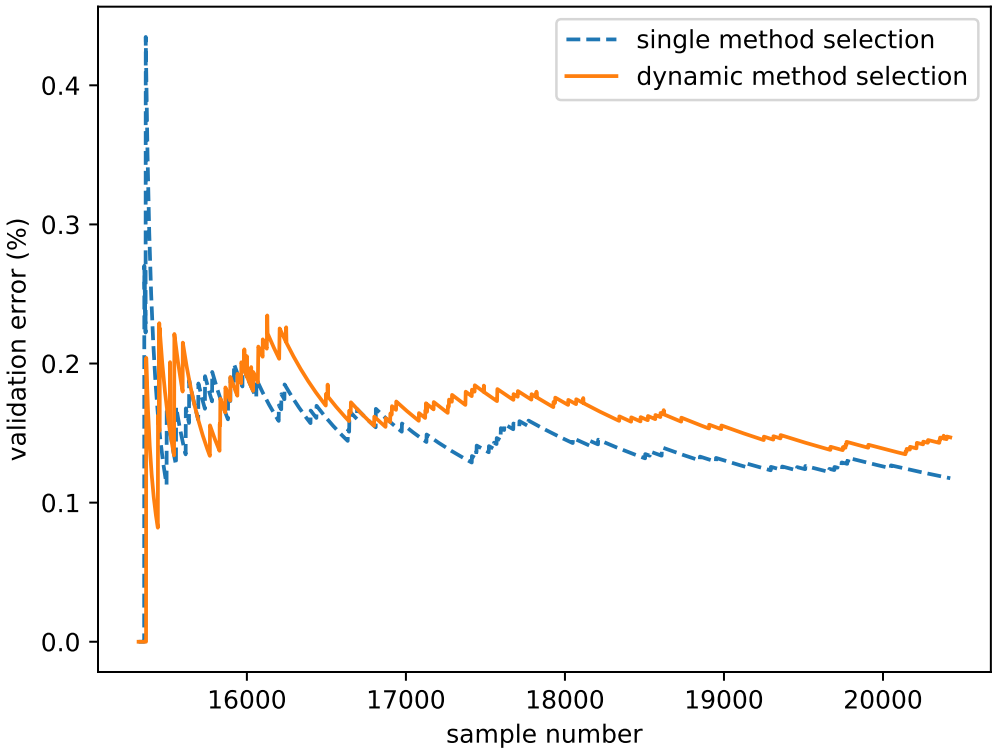


Figure 4-4: Plot of validation accuracy over time for the arithmetic data with 25% of samples used for validation.

is most ideal for the dynamic selection algorithm. The apparent lack of substantial accuracy improvements, however, warrants further discussion from the perspective of anticipated accuracy per data sample.

Finally, Figs. 4-4 through 4-6 provide each algorithm's validation error per sample for each dataset. Here, the x axis corresponds to the data sample number given that the first 75% of the data was used in training, while the y axis is the validation error, which is simply the validation accuracy subtracted from one. The error value at each time step is the result of every sample evaluated up to that point. Beginning with the arithmetic dataset plotted in Fig. 4-4, both algorithms to appear to essentially trade places repeatedly on which one provides the lowest amount of error. The near equal values provided in Tables 4.3 and 4.4 are likely the result of a balance between

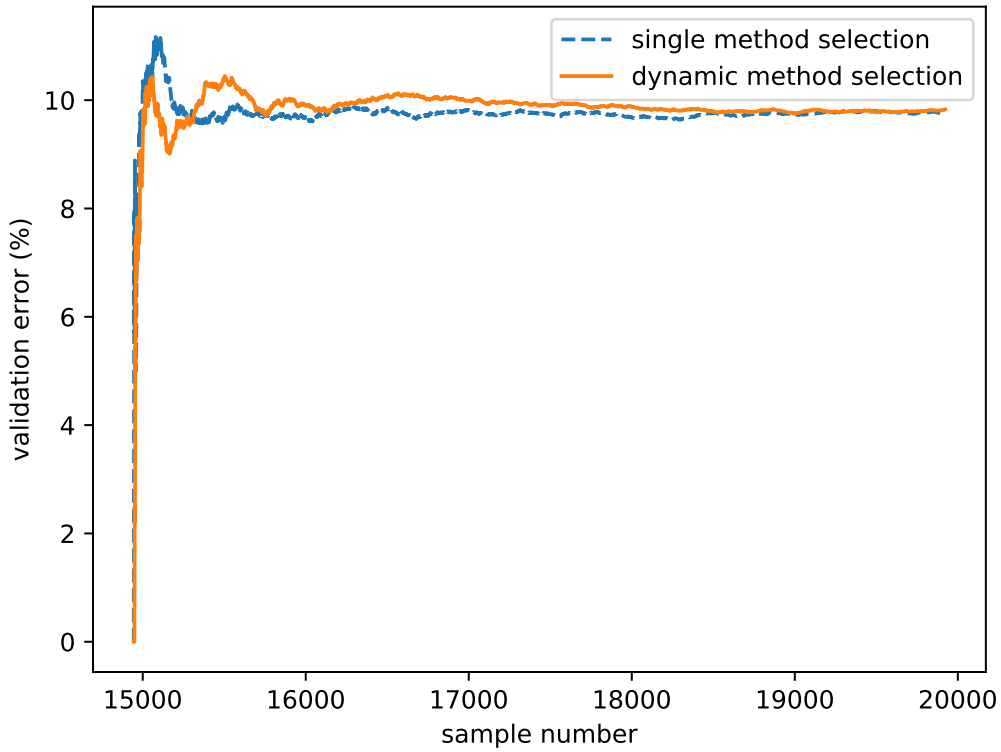


Figure 4-5: Plot of validation accuracy over time for the visual matching data with 25% of samples used for validation.

the substantial lead from the dynamic selection algorithm at an early data sample and the narrow but consistent lead by the single selection algorithm throughout the latter two thirds of the x axis.

A similar pattern is seen with the visual matching data in Fig. 4-5, including a brief significant lead by dynamic selection at the beginning followed by a narrow but consistent lead by single selection in the long term. The difference here, however, is that there are some brief crossover points towards the end of the plot. In any case, the end values once again balance out to being near equal as given in Tables 4.3 and 4.4. Fig. 4-6, however provides a very clear representation that dynamic selection outperforms single selection on the SAPHYRE data. With exception of a few points at the beginning of the plot, the dynamic error is consistently below that of the single

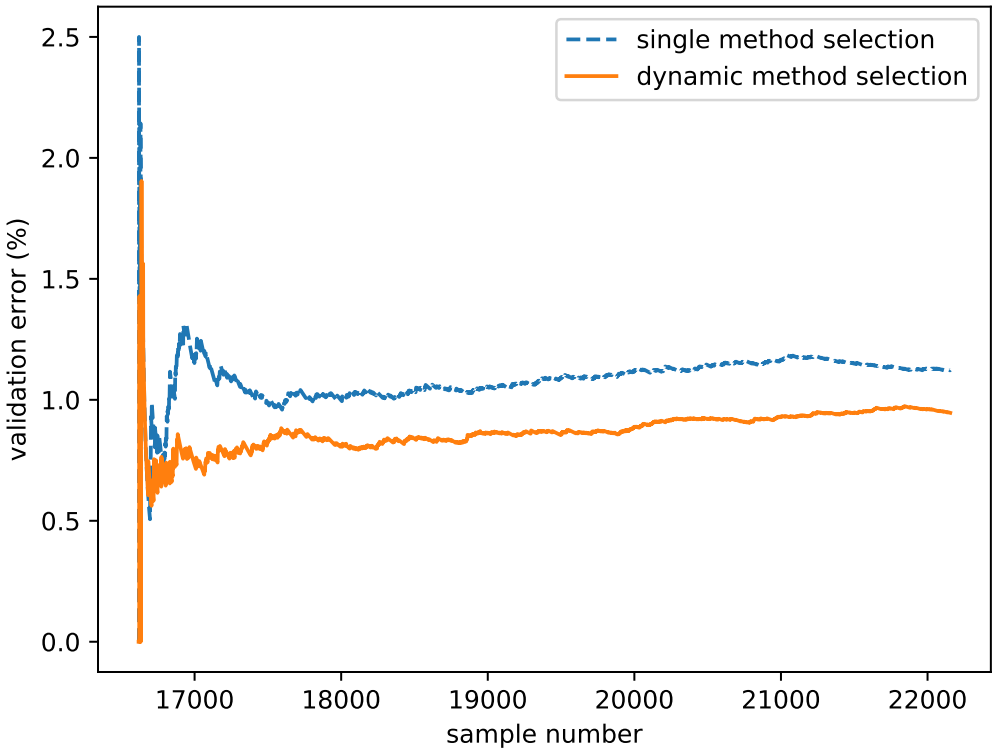


Figure 4-6: Plot of validation accuracy over time for the SAPHYRE data with 25% of samples used for validation.

algorithm. With these new observations intact, it is still beneficial to explore yet another dimension of results.

4.2.2 66% Validation

This subsection provides the results in a format similar to before but for the more unconventional validation ratio of 66%. This begins with single selection method results in Table 4.5. From there, Table 4.6 provides the results for the dynamic selection method. When comparing the validation errors to their 25% validation counterparts, a key observation is that the arithmetic data handles the training sample decrease and validation sample increase quite well with only a tenth of a percent decrease and is thus still in the optimal range of validation accuracy given in Table

Table 4.5: Summary of training accuracy, validation accuracy, and computational runtime under the single selection algorithm with 66% of samples used for validation.

Metric	Arithmetic	Visual Matching	SAPHYRE
Training Accuracy	.999	.956	.976
Validation Accuracy	.998	.817	.974
Runtime (s)	90.9	238	778

Table 4.6: Summary of training accuracy, validation accuracy, and computational runtime under the dynamic selection algorithm with 66% of samples used for validation.

Metric	Arithmetic	Visual Matching	SAPHYRE
RF Train. Accuracy	.999	.828	.976
DT Train. Accuracy	.999	.952	.984
ANN Train. Accuracy	.450	.717	.155
Validation Accuracy	.998	.799	.981
Runtime (s)	595	877	1248

3.3. The other two datasets contain a more noticeable drop in accuracy while still retaining some stability despite the change in number of samples. The SAPHYRE data holds on to these change quite well, while the visual matching data is more prone to change. In terms of training accuracy, the values in the arithmetic data stay roughly the same, while the other two datasets have slight decreases that are roughly equal to one another. As expected, runtime increases significantly in each case. When comparing values between the two tables of this subsection, the arithmetic data still retains accuracy values that are nearly equal to one another. In the visual matching data, however, the accuracy actually decreases in the dynamic method, while the SAPHYRE data once again maintains an increased accuracy in such a method.

Finally, each algorithm's validation error results over each sample for each dataset are given in Figs. 4-7 through 4-9. For the arithmetic and visual matching data in

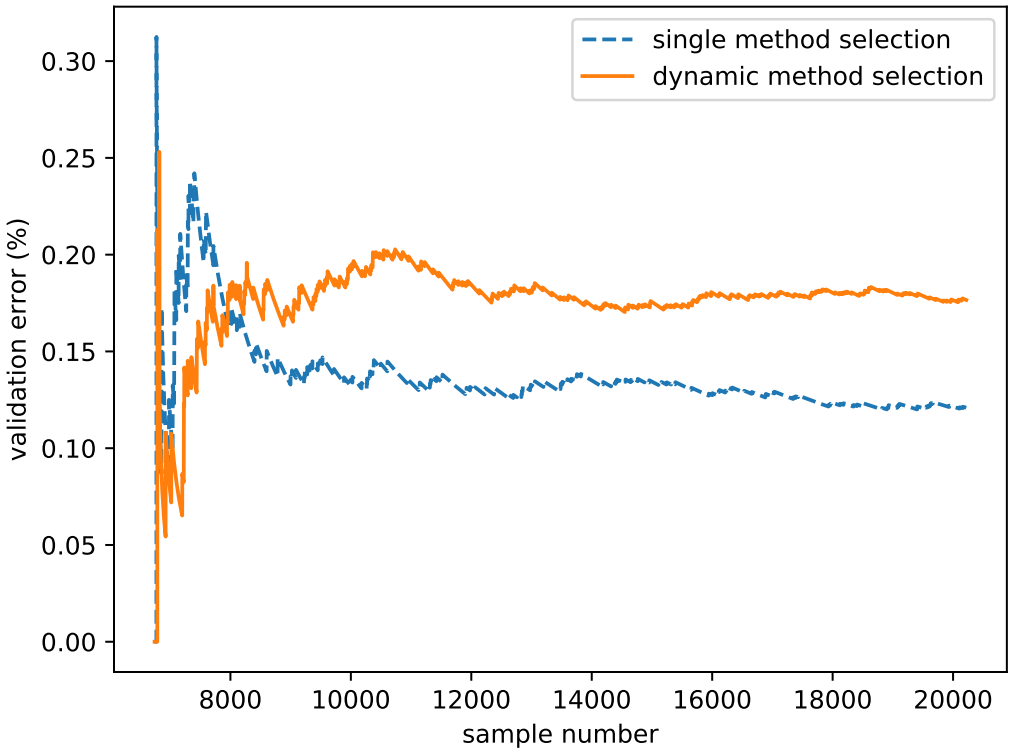


Figure 4-7: Plot of validation accuracy over time for the arithmetic data with 66% of samples used for validation.

Figs. 4-7 and 4-8, the trends remain largely the same as in Fig. 4-4, as there is a brief but sharp lead in dynamic accuracy followed later by a small but consistent lead by single method accuracy. Fig. 4-9 meanwhile continues the trend of dynamic accuracy maintaining a dominant for nearly the entire duration of the plot. The only fundamental differences between the latter two and their 25% validation counterparts is the numerical jump in error caused by the increase in validation data.

4.3 Conclusion

This chapter extended upon the comparative analysis made in Chapter 3 by providing two dynamic algorithms for dynamically selecting a machine learning technique

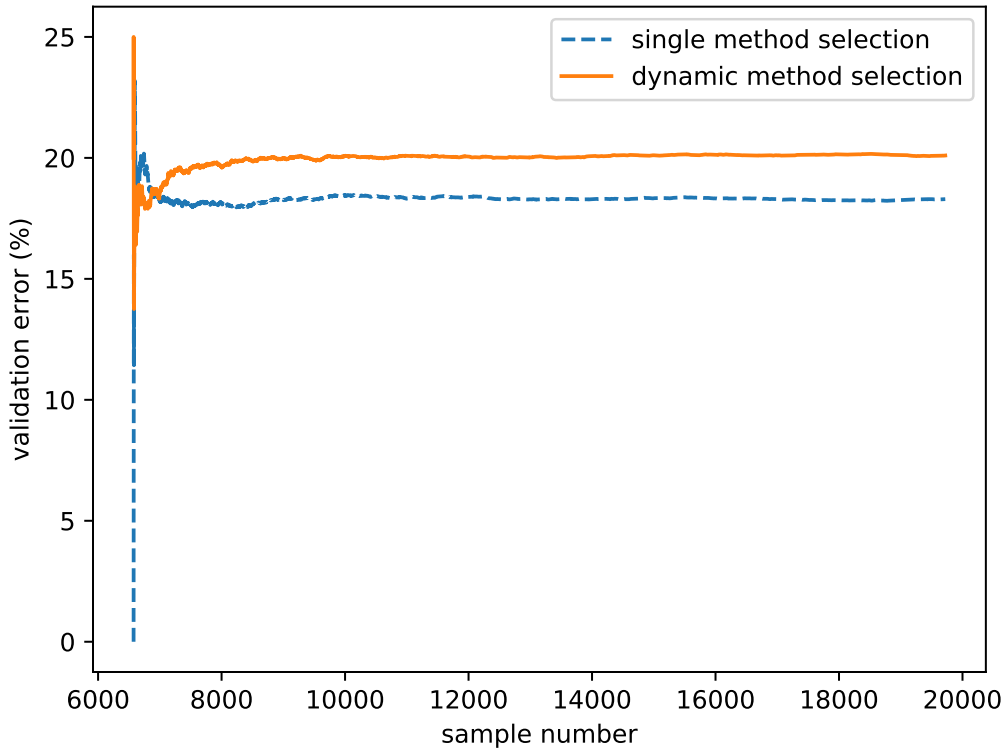


Figure 4-8: Plot of validation accuracy over time for the visual matching data with 66% of samples used for validation.

to a given dataset. The first was a static algorithm that selected a method based on the attributes of the given data. The second was a dynamic algorithm that trained all candidate ML techniques, selected an initial method in a manner similar to before and then switched techniques when the comparative accuracy of the present method was insufficient. Overall, the advantages and disadvantages of one algorithm over another comes down to the dataset involved.

While the arithmetic and visual matching data produced better results under the single selection algorithm, the SAPHYRE data showed significant accuracy improvement under dynamic selection. It produces over 99% accuracy when training data to validation data is a 2:1 ratio and 98% accuracy at an unconventional 1:2 ratio. Given that this dataset is the most relevant to the overall topic of aviation, this makes the

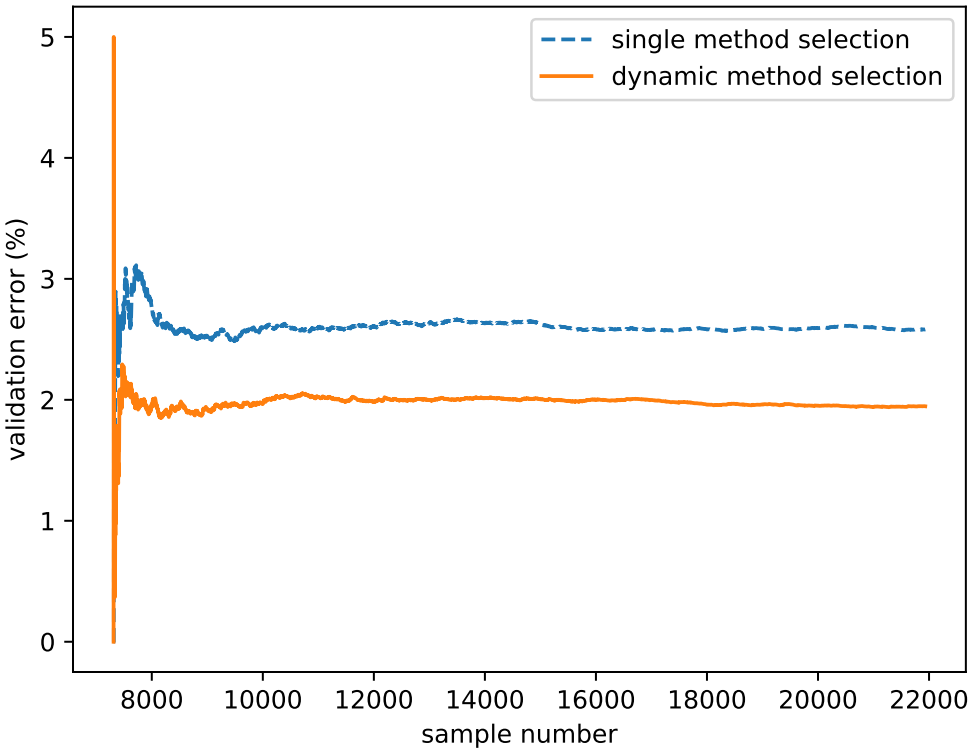


Figure 4-9: Plot of validation accuracy over time for the SAPHYRE data with 66% of samples used for validation.

second proposed algorithm as a significant milestone in the continuous improvement of manned flight safety. After comparing machine learning techniques and incorporating ML method selection in a new and novel algorithm, the next logical steps are to engage in similar processes for unmanned flight safety.

Chapter 5

Performance Assessment of Unmanned Aerial Vehicles

This next phase of the research involves the more recently established concept of unmanned aerial vehicles (UAVs). More specifically, this chapter begins a look into the technical realms of unmanned flight safety. While human pilots and cognitive workload can still play a part in this field when significant remote piloting is involved, this focus is primarily on autonomous UAVs, in which a remote pilot focuses more on enforcing broad mission objectives rather than controlling a vehicle's entire trajectory. One of the technical goals of this chapter is to enhance the prediction of UAV trajectory data by fusing past navigation data with other sensory data. A subsequent goal is then to compare the results of this new and novel method with multiple established prediction algorithms relevant to UAV applications. Overall, the results indicate that the inclusion of this type of data fusion provides significant advantages in UAV data prediction and outperforms the established methods in many different cases.

5.1 Introduction

In recent years, unmanned aerial vehicles (UAVs) have become a dominant force in military operations as well as in commercial and recreational applications. UAVs can be controlled either manually via a remote pilot or autonomously. Either case presents numerous degrees of variability and unpredictability due to the behavior of the human pilot or autonomous system and can adversely affect team performance [125], leading to the necessity of blame assignment [126]. Thus, creating a mechanism for explaining the behavior and decisions of autonomous UAVs is of particular interest, due to the novel factors that comprise such an explanation. These can include memory-based data structures, memory reconstruction, explanation algorithms, and explanation interfaces.

Episodic memory is a valuable tool that can substantially aid in many of these factors. It is particularly useful due to the ability to determine a quality of interestingness in intelligent agents [123, 136–139] as well as in other applications such as image processing [140, 141]. In a simplified context, interestingness can be expressed as a significant difference between a predicted result and an actual result. While correlating and predicting decisions based on sensory data [145] is already a fundamental and well-established concept, doing so in the context of finding interestingness to explain UAV behavior forms a new realm of possibilities.

In order to generate more robust and more useful predictions for such a context, optimal success stems from integrating larger amounts and greater varieties of input data. While integrating the former is straightforward (i.e. obtaining as many data samples as possible), forming an accurate prediction mechanism by including multiple types of dissimilar data becomes a challenge of great importance. When combining multiple types of data, predictions and classifications can be formed in a more comprehensive and more effective manner. For instance, when analyzing UAV

navigational data, a deeper understanding can be gained of the paths taken when factoring in weather data (e.g. a path is altered by strong winds) and/or camera or radar data (e.g. an enemy drone causes the UAV to retreat out of sight).

Two well-known approaches for data fusion include Bayesian Inference and Dempster-Shafer Evidence Theory (DSET), the latter of which is desired here due to its distinct capability of handling the concept of uncertainty [215], or confidence, in many UAV-based applications. Existing applications include a tracking and identification mechanism for aerial sensor networks in UAVs that uses DSET to account for the possibilities of varying confidence levels for sensor targets [216]. Another example has involved both DSET and Bayesian Inference for multi-UAV cooperative search, which found that the former approach is better able to find a desirable number of targets when sensor accuracy is low and the resultant data is more contradictory [217].

To address and compare the usefulness of DSET as a data fusion technique in UAV data prediction, this chapter presents the results for four machine learning techniques for predicting a UAV's location and trajectory based on the modeling of prior navigation data and radar data. One is a simple artificial neural network (ANN) trained with ant colony optimization that inputs radar data and a time step and outputs navigation-based attributes, including latitude, longitude, altitude, and heading. This follows a typical machine learning approach to data modeling, by partitioning a portion of the data samples for training and the remainder for validation. The second and third techniques are based on an extended Kalman filter (EKF) and unscented Kalman filter (UKF), respectively, which utilize dynamic UAV model equations. The fourth technique is a new approach proposed in this chapter that takes the same prior inputs and outputs, coupled with validation inputs, and fuses this information together using Dempster-Shafer Evidence Theory. This approach continues to use an ANN for predicting altitude and heading but incorporates DSET for the estimation of latitude and longitude. Thus, it is able to forego a training stage for the latter

outputs and make better use of the dissimilarities among data types. Overall, the goal of these methods is to clearly demonstrate the usefulness of this type of data fusion in enhancing machine learning for UAV data prediction.

When forming prediction mechanisms to accurately model UAV behavior, several questions must be asked in order to fully examine and utilize the usefulness of the proposed data fusion technique, such as

1. How far ahead can a prediction be made with reasonable accuracy?

2. How much more does an algorithm benefit with data fusion than without?

3. How dissimilar can the fused types of data be while still producing stable results?

To address these questions, a novel dataset was obtained that tracks a wide range of sensory data from six UAVs over the course of 695 seconds, eventually leading to a crash. This includes attributes such as elapsed time, mission objectives, and navigational data, the latter of which is of particular interest in this research. The context of this data provides a concrete example of potential anomalies between expectations and outcomes as well as the aftermath that can follow, thereby necessitating a role for explainable behavior.

5.2 Methodology

The first algorithm, to be used as a basis of comparison, follows a typical machine learning approach. A UAV's navigational coordinates (e.g. latitude, longitude, heading, altitude) serve in conjunction with a time step as well as its radar data, particularly its distances from other items of interest, as inputs and outputs to an ANN. The overall goal of this algorithm is to determine navigational coordinates for some future point in time. The ability to accurately determine multiple future positions over a period of time unlocks the potential for predictive path planning and

more meaningful autonomous behavior explanation. While the ANN is able to interpret a variety of input and output data through the use of data scaling, no explicit data fusion takes place. The second and third algorithms are based on continuous time equations for a UAV given by [208]. These are applied on a step by step basis, transforming each of the UAV coordinates per time step in the process.

The proposed algorithm, however, follows a specific data fusion approach that encompasses five major steps: (1) Conversion of latitudinal and longitudinal coordinates into a distance from a common point of origin, (2) Resulting distances fused with radar-based distances from other UAVs as basic probability assignments (BPAs) for Dempster-Shafer (DS) based calculation, (3) Prediction of future distance from origin based on the DS plausibility function. (4) Heading and altitude given as inputs to a smaller, simpler ANN to predict future instances of the same two values, and (5) Conversion back to Cartesian coordinates using DSET-based magnitude and ANN-based heading.

5.2.1 Multi-output Prediction using Artificial Neural Networks

An artificial neural network (ANN) is a well-known statistical algorithm that has widespread familiarity in the realms of machine learning and data prediction. Due to their flexiblity and longevity of usage throughout the realm of machine learning, ANNs are used here as a basic data prediction mechanism for UAV location prediction. This will contrast greatly to the DSET-based data fusion approach proposed in the subsequent subsection in an effort to demonstrate some eventual and clear advantages that fusion of dissimilar data can have on UAV data prediction.

The proposed ANN will follow a basic multilayer perceptron (MLP) structure, similarly to the previous two chapters. Fig. 5-1 depicts an overview of the inputs and outputs given in this context. The problem formulation in this case is presented as: *Given a set of radar data and a time step, what will be heading, altitude, and*

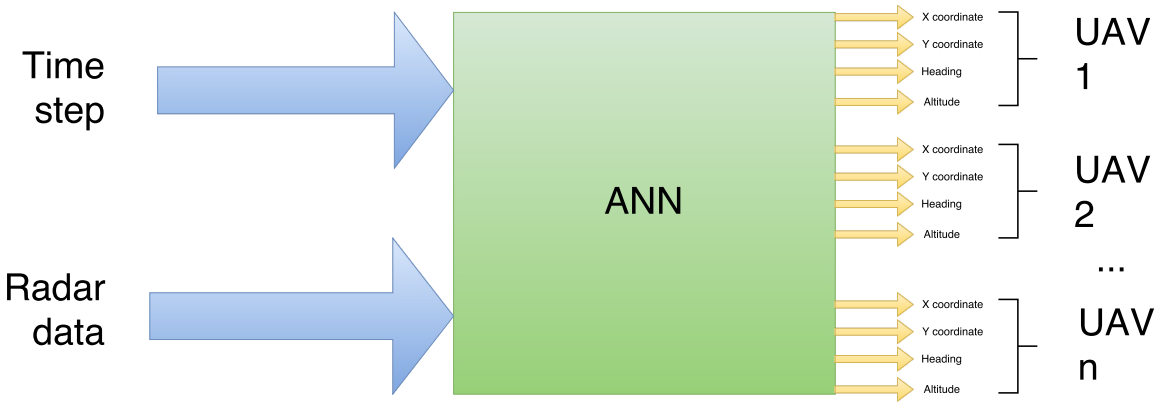


Figure 5-1: Overview of inputs and outputs in ANN algorithm.

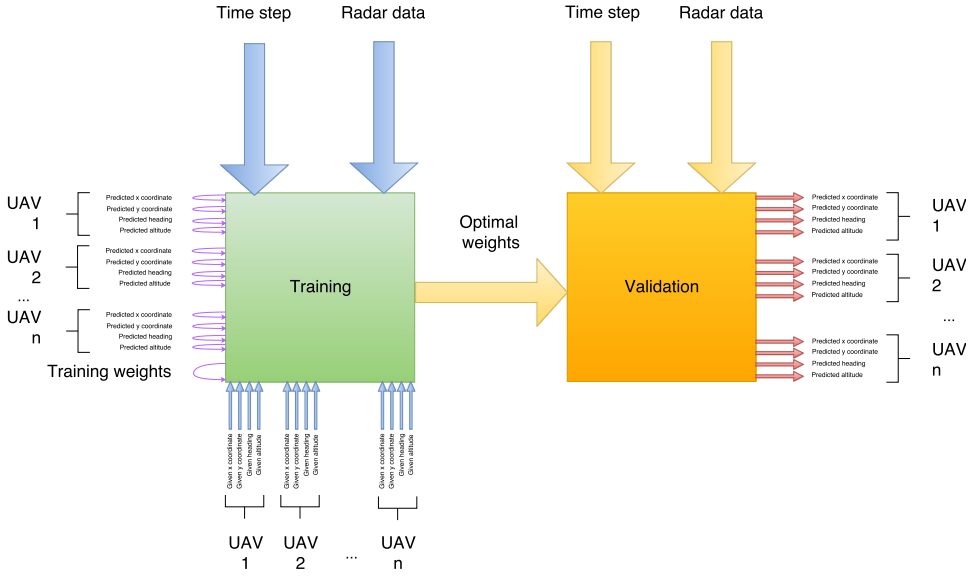


Figure 5-2: Detailed ANN algorithm.

Cartesian coordinates of one or more UAVs? The proposed algorithms are designed such that one can make predictions regarding one UAV of interest, multiple UAVs, or even an entire network thereof. Hence, Fig. 5-1 gives the output values in the general case of n UAVs, where n can be any positive integer. Fig. 5-2 depicts the complete two-stage process.

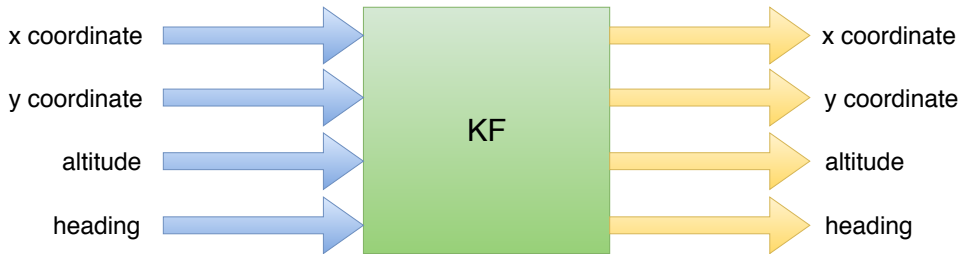


Figure 5-3: Overview of Kalman filter algorithm.

5.2.2 State-based Prediction Based on Kalman Filters

The Kalman filter based models are derived from the dynamic model equations for a UAV given in [208]. For the longitudinal coordinate, x, the equation is given by

$$x_{t+1} = x_t + v(t)cos(\theta(t)), \tag{5.1}$$

where θ is the heading and $v(t)$ is the ground speed of the UAV. Because extensive use of Kalman filter is beyond the scope of this research and because the following assumption still produces competitive results, v(t) is assumed to be a value of one in these equations. Similarly, the latitudinal coordinate, y, is given by

$$y_{t+1} = y_t + v(t)sin(\theta(t)). \tag{5.2}$$

For reasons similar to those of the prior given assumption, the equations for heading and altitude are simply given identity functions of

$$\theta_{t+1} = \theta_t \tag{5.3}$$

and

$$h_{t+1} = h_t. \tag{5.4}$$

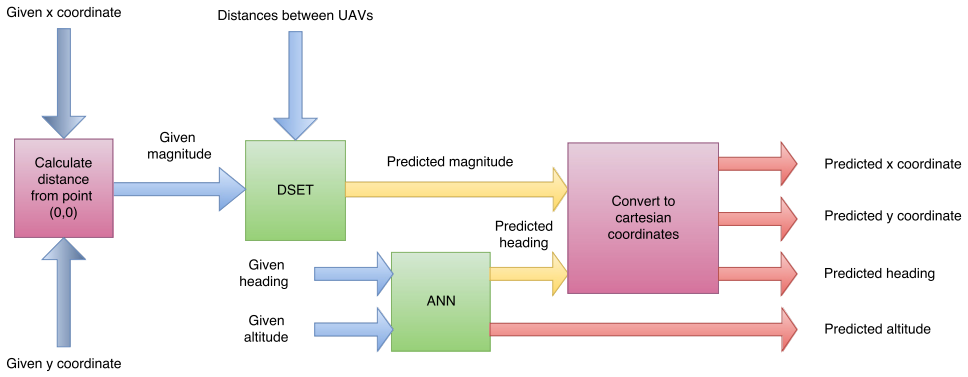


Figure 5-4: Detailed ANN+DSET algorithm.

Together, these equations are evaluated over every sample of training and validation data. In addition, an EKF or UKF is applied with the equations during the training stage. Fig. 5-3 provides an overview of the algorithm for both EKF and UKF. In this context, the only substantial differences between the two filters lie within the built-in MATLAB functions that are utilized.

5.2.3 Fusion of Navigational and Radar Data for Prediction of Magnitude

Recall from Subsection 2.5.3 that Dempster-Shafer Evidence Theory can be viewed as an innovative generalization of Bayesian probability theory, diverging from Bayesian theory based on the handling of ignorance [215], or uncertainty. Bayesian approaches typically assume zero ignorance, or 100% confidence, for any probabilistic value. For internal probability factors, this approach is usually valid. DSET, however, is unique in that it uses external factors. Thus, one can no longer automatically assume this kind of condition. For that reason, an array of evidence factors is utilized to determine all possible outcomes in a given scenario (e.g. past and present distances between a UAV and some point of origin).

5.2.3.1 Set Representation

Given that a basic probability assignment (BPA) carries three distinct properties (the lower bound $\underline{x}$, the upper bound $\bar{x}$, and the degree of confidence μ), a BPA can be represented as a set of these properties for the sake of organization and efficiency. In such a case, the resultant BPA m would be defined as [201]

$$m = \{\underline{x}, \bar{x}, \mu\}. \tag{5.5}$$

Calculation of belief and plausibility, however, requires multiple BPAs, or the cumulative distribution function (CDF) of an aggregate BPA. Therefore, representing the individual assignments as a set of sets is also beneficial. $m_1^u, m_2^u, \ldots, m_n^u$ can represent all BPAs (of n quantity) in Θ as [201]

$$m = \begin{Bmatrix} m_1^u \\ m_2^u \\ \vdots \\ m_n^u \end{Bmatrix} = \begin{Bmatrix} \{\underline{x}_1^u, \bar{x}_1^u, \mu_1^u\} \\ \{\underline{x}_2^u, \bar{x}_2^u, \mu_2^u\} \\ \vdots \\ \{\underline{x}_n^u, \bar{x}_n^u, \mu_n^u\} \end{Bmatrix} \tag{5.6}$$

and $\bar{x}_1^u < \bar{x}_2^u < \cdots < \bar{x}_n^u$. The latter expression is especially important, as this sorted order of upper bounds can be manipulated to define belief from a specific set-based context. Suppose the superscript u indicates that the contents of m are arranged in order of increasing upper bound. In addition, μ is the degree of confidence in each

BPA and therefore can be denoted as $\mu_1^u, \mu_2^u, \ldots, \mu_n^u$. Then, belief is expressed as [201]

$$Bel = \begin{Bmatrix} \{\bar{x}_1^u, \mu_{1,new}^u\} \\ \{\bar{x}_2^u, \mu_{2,new}^u\} \\ \{\bar{x}_3^u, \mu_{3,new}^u\} \\ \vdots \\ \{\bar{x}_n^u, \mu_{n,new}^u\} \end{Bmatrix}, \tag{5.7}$$

where

$$\mu_{1,new}^u = \mu_1^u \tag{5.8}$$

and

$$\mu_{n,new}^u = \sum_{i=1}^{n-1} \mu_{i,new}^u. \tag{5.9}$$

Now, m can be rearranged and $m_1^u, m_2^u, \ldots, m_n^u$ renamed so that [201]

$$m = \begin{Bmatrix} m_1^l \\ m_2^l \\ \vdots \\ m_n^l \end{Bmatrix} = \begin{Bmatrix} \{\underline{x}_1^l, \bar{x}_1^l, \mu_1^l\} \\ \{\underline{x}_2^l, \bar{x}_2^l, \mu_2^l\} \\ \vdots \\ \{\underline{x}_n^l, \bar{x}_n^l, \mu_n^l\} \end{Bmatrix}, \tag{5.10}$$

where $\underline{x}_1^l < \underline{x}_2^l < \cdots < \underline{x}_n^l$ and the superscript l indicates that the contents of m are arranged in order of increasing lower bound. From here, all conditions are satisfied

to define plausibility as [201]

$$Pl = \left\{ \begin{array}{c} \{\bar{x}_1^l, \mu_{1,new}^l\} \\ \{\bar{x}_2^l, \mu_{2,new}^l\} \\ \{\bar{x}_3^l, \mu_{3,new}^l\} \\ \vdots \\ \{\bar{x}_n^l, \mu_{n,new}^l\} \end{array} \right\}, \tag{5.11}$$

where

$$\mu_{1,new}^l = \mu_1^l \tag{5.12}$$

and

$$\mu_{n,new}^l = \sum_{i=1}^{n-1} \mu_{i,new}^l. \tag{5.13}$$

To summarize, the bottom and top rows respectively represent the most and least believable outcomes. The latter is of greatest importance, where $\bar{x}_n^u$ is the most believable outcome value and $\mu_{n,new}^u$ is the belief probability, which is always 1, or 100%, given that it is the sum of every confidence value. Likewise, the bottom and top rows of the belief set respectively denote the most and least plausible outcomes. In this case, the most believable row is most significant, as $\underline{x}_n^l$ represents the most plausible outcome and $\mu_{n,new}^l$ is also 100% probability. Recent successes in utilizing DSET has involved the use of the plausibility function [201], which therefore serves as a core theoretical component to the proposed algorithm.

5.2.3.2 Proposed Representation of Distance Range as BPA

To effectively fuse together any of the dissimilar measurement types, all measurements must be converted to a common value type. Such an ideal value in this application is distance. This can stem from either a UAV's position, measured as

the distance from a point of origin to said position, or from radar data denoting the distance of an object of interest from the UAV. The next step is to find distance values from external measurements that lie in the corresponding range. After filtering these values, the minimum and maximum become the lower and upper bounds of the resulting BPA. Because the BPAs will aggregate later, the confidence value is given as 1 for now. Thus, from Equation 5.5,

$$m_i = \{d_{min}, d_{max}, 1\} \tag{5.14}$$

can be obtained, where d_{min} is the minimum distance value and d_{max} is the maximum.

Once every BPA is formed, aggregation can occur. Manually adding new BPAs can often be a highly inefficient and cumbersome process, in which case an ideal alternate plan is to sample from a distribution. Also known as discretization, this process generates n discrete samples from a lesser number of initialized BPAs, which then collectively form a CDF of BPA structures [199]. Therefore, a solution to smoothen the transition between BPAs is to sample each m_i by a function handle such as inverse normalization. After sampling, the aggregate BPA is now an adimensional, unitless entity, as is required for mass assignments. Now each BPA can fit the parameters of Equation 5.5 and be written as

$$m_i = \{\underline{x}, \bar{x}, \frac{1}{n_s n_v}\}, \tag{5.15}$$

where $1/(n_s n_v)$ denotes an evenly distributed confidence probability of one over the total number of samples and total number of variables per sample, respectively. Distances can be one-dimensional, such as from a UAV's longitudinal coordinate to a vertical point at x = 0, from a latitudinal coordinate to a horizontal point at y = 0, or from an altitude coordinate to a ground-based point at z = 0. They can also be two dimensional, such as the radar distances from other UAVs. Thus, the set of all

possible distances is expressed as

$$\lambda = \{\lambda_x, \lambda_y, \lambda_h, \lambda_{r,1}, \lambda_{r,2}, \ldots, \lambda_{r,p}\}, \tag{5.16}$$

where p is the number of neighboring UAVs, r is the radar category of measurements,

$$\lambda_x = \begin{Bmatrix} m_{1,x} \\ m_{2,x} \\ \vdots \\ m_{n,x} \end{Bmatrix}, \lambda_y = \begin{Bmatrix} m_{1,y} \\ m_{2,y} \\ \vdots \\ m_{n,y} \end{Bmatrix}, \lambda_h = \begin{Bmatrix} m_{1,h} \\ m_{2,h} \\ \vdots \\ m_{n,h} \end{Bmatrix}, \tag{5.17}$$

and

$$\lambda_{r,1} = \begin{Bmatrix} m_{1,r,1} \\ m_{2,r,1} \\ \vdots \\ m_{n,r,1} \end{Bmatrix}, \lambda_{r,2} = \begin{Bmatrix} m_{1,r,2} \\ m_{2,r,2} \\ \vdots \\ m_{n,r,2} \end{Bmatrix}, \ldots, \lambda_{r,p} = \begin{Bmatrix} m_{1,r,p} \\ m_{2,r,p} \\ \vdots \\ m_{n,r,p} \end{Bmatrix}. \tag{5.18}$$

With all aggregation intact, the next step is the actual data prediction process.

5.3 Methods and Materials

The ANN and ANN+DSET algorithms were programmed using the pseudocode in Figs. 5-5 and 5-6. The former details the ANN approach using ant colony optimization, while the latter describes the DSET-based data fusion approach. For the ANN, the pseudocode only details the training process, as validation simply applies validation inputs into the trained neural network model with optimal weights. The ANN+DSET pseudocode, however, involves the validation process after an ANN for heading and altitude has already been trained.

For evaluation, the primary concern is with prediction accuracy. By finding notice-

```
train_input ← read time and radar data for all training time steps
train_output ← read x, y, h, and θ for all training time steps
Initialize pheromone values
Create ants
while termination condition has not been reached do
   for each ant do
      Construct an ant solution
      Evaluate ant's fitness
   end for
   for all pheromone values do
      Decrease pheromone (evaporation)
   end for
   for all good solutions do
      Increase pheromone
   end for
end while
return  optimal training weights
```

Figure 5-5: Pseudocode of ANN algorithm trained with ACO.

```
train_input ← read time and radar data for all training time steps
train_output ← read x, y, h, and θ for all training time steps
val_input ← read x, y, h, and θ for current validation time step
theta ← read current theta value obtained from ANN
for all neighboring UAVs (all input types except time step) do
   distances ← all radar values of a particular neighboring UAV over time
   BPA_per_input ← min_distance, max_distance, 1
   apply sampling to each input BPA
end for
for all coordinate types (all output types) do
   coord_values ← all values of a particular coordinate type over time
   BPA_per_output ← min_coord_value, max_coord_value, 1
   apply sampling to each ouput BPA
end for
aggregate all BPAs
pl_dist ← most plausible distance
pred_dist ← pl_dist * (current_time - max_training_time)
delta_x , delta_y ← theta and pred_dist as Cartesian coordinates
new_x, new_y ← x + delta_x, y + delta_y
return  theta, new_x, new_y
```

Figure 5-6: Pseudocode of ANN+DSET algorithm.

```
train_in, val_in, train_out, val_out ← all possible input and output data
output ← all possible accuracy and coordinate data
for all trials do
  for all time partitions do
    for all validation ratios do
      for all radar data restrictions do
        for individual UAVs versus all UAVs combined do
          train first ANN using x, y, h, and θ in outputs
          train second ANN using only h and theta in outputs
          for all training time steps do
            apply EKF filter
            apply UKF filter
          end for
          for all validation time steps do
            ann_out_1 ← predicted x, y, h, and θ of first ANN
            ann_out_2 ← predicted h and θ of second ANN
            for all UAVs (1 if evaulating all UAVs together) do
              dset_out ← predicted x and y using DSET
              store all coordinate data in output
            end for
            ekf_out_1 ← predicted x, y, h, and θ
            ukf_out_1 ← predicted x, y, h, and θ
            ann_error ← ANN error using RMSE
            dset_error ← ANN+DSET error using distance formula
            ekf_error ← EKF error using RSME
            ukf_error ← UKF error using RSME
            store error data in output
          end for
        end for
      end for
    end for
  end for
end for
average output over all trials
return output
```

Figure 5-7: Pseudocode of complete evaluation process.

able discrepancies between a predicted value and an actual value, significant amounts of interestingness can be determined that can later assist in forming explainable behavior when a UAV performs a task unsuccessfully. To provide a thorough and comprehensive analysis, the algorithms were tested across seven different dimensions of variables, consisting of:

1. Method (ANN, ANN+DSET, EKF, UKF)
2. Time
3. Scenario (as detailed in Table 5.1)
4. Individual UAV and all UAVs combined
5. Radar restrictions (UAVs can be detected within 100, 10, and 1 meters)
6. Percent validation (25 and 66 %)
7. Trial number

Fig. 5-7 details the complete evaluation process. The end result was a single MATLAB variable in the form of a cell array. Each cell corresponded to each scenario and consisted of a six-dimensional matrix. The dimensions were time, radar restrictions, data type, UAV number, one UAV versus all, and trial number, respectively. The experiment completed by averaging each matrix across all trials, thereby resulting in a cell array of five-dimensional matrices.

5.3.1 Data Acquisition

To provide a practical, real world direction in our algorithms, a novel dataset was obtained that follows six UAVs over the course of over 695 seconds, eventually leading

Table 5.1: Summary of timeline partitions.

Scenario	Start (s)	End (s)	Significance of ending
1	1	140	UAV contacts temporarily lost
2	141	180	Added new goal
3	181	540	UAV heading different from expected
4	541	575	UAV heading different from expected
5	576	678	Crash

to a crash. The goal of the data usage is to examine different points of interestingness, based on factors such as mission changes, technical challenges (e.g. lost communication, crashes), and discrepancies between predictions and results. While making predictions over the full timeline can be useful, partitioning periods of times based on significant, or interesting, events can help to provide a more thorough analysis of each algorithm's effectiveness. This is primarily due to the nature of episodic memory, as separating log data into temporal bands is one of the hallmarks of the computational episodic memory approach. Thus, after extensive observations involving a simulation of the data, it has been established that algorithms will be evaluated on five distinct time partitions as well as on the full timeline of events. Table 5.1 provides a summary of the partitions selected as well as the reasons for such selections.

5.3.2 Evaluation Metrics

Many different metrics are currently available for evaluating prediction error in an ANN. One of the most common includes mean squared error (MSE), depicted as

$$E_{MS} = \frac{1}{n}\sum_{i=1}^{m}\sum_{j=1}^{n}(y_{ij,act} - y_{ij,pred})^2, \tag{5.19}$$

where m is the number of samples, n is the number of outputs, and y is the output, denoted as either actual or predicted. Another is root mean squared error (RMSE),

a minor revision of the former, which is given as

$$E_{RMS} = \frac{1}{n}\sum_{i=1}^{m}\sqrt{\sum_{j=1}^{n}(y_{ij,act} - y_{ij,pred})^2}. \tag{5.20}$$

In addition, error can be applied to be more specific to the application of UAV prediction. Because the desired measurement is simply the difference between expected coordinates and actual coordinates, error can be perceived in terms of distance, which can therefore be given as

$$E_{dist} = \sqrt{(x_{act} - x_{pred})^2 + (y_{act} - y_{pred})^2 + (h_{act} - h_{pred})^2} \tag{5.21}$$

when considering the UAV in three dimensions. In this case, x is the longitudinal coordinate, y is the latitudinal coordinate, and h is the altitude. Given that this distance formula is simply the square root of the sum of the outputs, conclusions can be drawn that it is essentially an application of root mean square error. From here, for the sake of consistency, RMSE can be applied as an evaluation method for prediction accuracy of all four methods. Although heading is also an output in both methods, it is not represented in the distance formula. Therefore, for the sake of consistency, error will only be evaluated for the x, y, and h outputs given that heading still plays an implicit role in determining x and y. Thus, the general error function can be rewritten as

$$E = \frac{1}{n}\sum_{i=1}^{m}\sqrt{(x_{i,act} - x_{i,pred})^2 + (y_{i,act} - y_{i,pred})^2 + (h_{i,act} - h_{i,pred})^2} \tag{5.22}$$

when averaging the error across all samples.

5.3.3 Experimental Settings

To verify and compare the effectiveness of both methods, a variety of simulation scenarios was conducted on the data described in Subsection 5.3.1 using the MATLAB scientific programming environment [218]. To provide built-in DSET functions for the purpose of this experiment, the open source Imprecise Propagation Probablity (IPP) Toolbox was acquired, released under the GNU General Public License (GPL), to address all DSET-based functions of our ANN+DSET algorithm. An overview of the default simulation parameters in the experimental setup is as follows:

1. Six UAVs deployed in a 100 meter by 100 meter environment over a time range of 695 seconds.

2. ANN trained with ant colony optimization (ACO) using ρ value of 0.01, 30 ants, 50 discrete points, and a maximum of 30 iterations

3. ANN model consisting of 7 hidden layers with 5 hidden neurons per layer

4. DSET descretization consisting of 10 samples as an inverse normalized distribution.

5. Experimental results as the average of 10 independent trials for consistency and reliability.

All trials and variations were conducted in MATLAB 2017a on a Windows 10 Professional desktop with an Intel Core i5-6600 3.30 GHz quad core CPU and 16 GB of RAM, bearing similar but not equal specifications to those in Chapters 3 and 4.

5.4 Experimental Results

In addition to obtaining numerical results as detailed in the previous subsection, analysis will be based on the following questions:

1. Under which conditions is the maximum near-term prediction accuracy achieved?

2. Under which conditions is extrapolation from UAV data more limited with DSET than without?

3. Under which conditions does near-term prediction accuracy remain more reliable with DSET (i.e. less prone to significant change) than without as test cases become increasingly complex?

In other words, the first question considers prediction error as a function of time and asks which method has a lower absolute minimum. While the main focus is on near-term prediction accuracy and thus error values at early time steps, a look at the entire error function is also under consideration. The second question involves cases in which less data is available or data becomes less certain. From there, the conditions are evaluated in which each method can and cannot obtain steady levels of accuracy. The third question, meanwhile, bears some similarity to the second, but here the goal is to determine whether increased restrictions or complexities in the data result in significant amounts of change in prediction accuracy. Answering these questions with careful consideration can greatly aid in portraying a full and beneficial analysis on the usefulness of DSET-based data fusion in this application.

5.4.1 Single-UAV Analysis

Table 5.2 provides a summary of the results for all four methods for each scenario and for each of the three validation ratios, which are again 25% and 66%. In ANN+DSET, most scenarios follow a predictable trend, in which the restriction of training data resulted in an increase in average error per time step. Scenarios 4 and 5, however, break from this trend, as the error actually decreases.

The ANN results, meanwhile, follow a more mixed set of patterns, as only scenario 4 shows an increase in error as the number of training samples decreases. In addition,

Table 5.2: Simulation results evaluating average prediction error over time by validation ratio, method, and scenario.

	25% validation				66% validation			
Scenario	ANN	DSET	EKF	UKF	ANN	DSET	EKF	UKF
1	.3019	.0242	.1683	.1683	.2825	.0603	.1600	.1600
2	.2774	.0100	.1884	.1884	.3155	.0200	.1731	.1731
3	.3325	.1416	.2191	.2191	.3208	.1802	.2270	.2270
4	.3194	.0358	.1955	.1955	.2786	.0277	.1963	.1964
5	.4110	.1043	.1859	.1859	.3640	.0819	.1987	.1988
Full	.3623	.1205	.2304	.2304	.3590	.2108	.2355	.1991

the reverse outcome appears in scenarios 1 and 5 as well as in the full timeline. 2 and 3, however, follow a more nonlinear pattern, with error at its lowest under 50% validation. EKF and UKF also produce mixed results by increasing under some scenarios and decreasing under others. A possible explanation for this new flexibility in validation ratios could be the fact that this algorithm does not shuffle the samples before splitting them into training and validation data. This is done in order to narrow the problem focus to only make future predictions given past and present data. Reverting to the second question based on validation ratios alone, the inclusion of DSET poses more restrictions on data extrapolation, given that ANN without DSET has mostly seen a decrease in prediction error under less conventional validation ratios than it has under the more typical 25%.

A noticeable trend in favor of the DSET component is the fact that validation accuracy is consistently lower with DSET than without. This trend holds true in Table 5.2 as well as in the next three tables. Tables 5.3 and 5.4 show the validation accuracy for each individual UAV by method and by scenario. The latter covers the Kalman filters, while former provides the results for ANN and ANN+DSET. Because the past table demonstrated relatively stable results at 66% accuracy, this ratio serves as the standard for all remaining tables (except Table 5.6) and figures throughout this

Table 5.3: Simulation results for ANN and ANN+DSET evaluating average prediction error over time for each individual UAV by method and by scenario.

	UAV 1		UAV 2		UAV 3		UAV 4		UAV 5		UAV 6	
Scenario	ANN	DSET	ANN	DSET	ANN	DSET	ANN	DSET	ANN	DSET	ANN	DSET
1	.2461	.0638	.2852	.0668	.2792	.0657	.2625	.0636	.3608	.0657	.2776	.0363
2	.2436	.0205	.4522	.0213	.3004	.0206	.2543	.0216	.3205	.0228	.3218	.0133
3	.4066	.2747	.3205	.2459	.3029	.1561	.3587	.1528	.2981	.1616	.2377	.0903
4	.2787	.0163	.2554	.0172	.2609	.0166	.2512	.0183	.2294	.0168	.4318	.0812
5	.4110	.1640	.3913	.0462	.2778	.0480	.4072	.1367	.3865	.0459	.3702	.0506
Full	.3026	.2427	.2933	.2273	.4527	.1687	.3748	.2294	.3774	.2339	.3533	.1630

Table 5.4: Simulation results for EKF and UKF evaluating average prediction error over time for each individual UAV by method and by scenario.

	UAV 1		UAV 2		UAV 3		UAV 4		UAV 5		UAV 6	
Scenario	EKF	UKF	EKF	UKF	EKF	UKF	EKF	UKF	EKF	UKF	EKF	UKF
1	.1693	.1693	.1700	.1700	.1725	.1725	.1705	.1705	.1699	.1699	.1081	.1081
2	.1746	.1746	.1774	.1774	.1865	.1865	.1767	.1767	.1781	.1781	.1453	.1454
3	.2174	.2174	.2233	.2233	.2456	.2456	.2925	.2925	.2308	.2308	.1524	.1524
4	.1817	.1817	.1789	.1790	.1735	.1736	.1783	.1783	.2234	.2236	.2420	.2420
5	.1768	.1768	.2139	.2139	.1691	.1691	.1862	.1862	.2890	.2891	.1576	.1576
Full	.2509	.2509	.2215	.2215	.2538	.2538	.2490	.2490	.2377	.2377	.2002	.2002

chapter. When interpreting these results, it is important to note in Table 5.2, the lowest amount of prediction error under this validation ratio occurred in scenario 1 under ANN, EKF, and UKF and in scenario 2 under ANN+DSET, while the highest error happened under the full timeline for all four algorithms.

For ANN+DSET, its strength in scenario 2 is only verified by UAV 6 alone, as it is the UAV to have the lowest error in this scenario, but it does so by a fairly wide margin. This is coupled by having the second lowest errors in all five of the other UAVs. Conversely, ANN's weakness in scenario 3 is only noticeable in UAVs 3 and 4, while ANN+DSET's high error is clearly visible in UAVs 3-6 and partly visible with second highest error values in the other two drones. While these results do not directly contribute to any of the three questions, it is important nonetheless to observe

Table 5.5: Simulation results evaluating average prediction error over time by radar capability, method, and scenario.

	100 meters				10 meters				1 meter			
Scenario	ANN	DSET	EKF	UKF	ANN	DSET	EKF	UKF	ANN	DSET	EKF	UKF
1	.2852	.0603	.1600	.1600	.2875	.0602	.1600	.1600	.3511	.0611	.1600	.1600
2	.3155	.0200	.1731	.1731	.2985	.0194	.1731	.1731	.2533	.0201	.1731	.1731
3	.3208	.1802	.2270	.2270	.3483	.1834	.2270	2270	.3475	.1806	.2270	.2270
4	.2786	.0277	.1963	.1964	.3258	.0279	.1963	.1964	.3100	.0280	.1963	.1964
5	.3740	.0819	.1987	.1988	.3638	.0810	.1987	.1988	.3202	.0816	.1987	.1988
Full	.3590	.2108	.2355	.2355	.3725	.2070	.2355	.2355	.3660	.2127	.2355	.2355

the behavior of the individual UAVs, given that all other results in this subsubsection are the averages of all the UAV's in terms of prediction and performance.

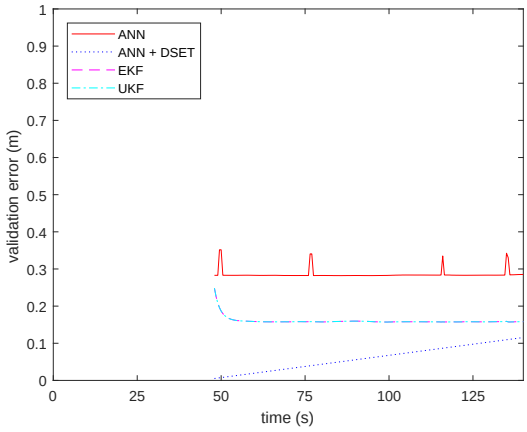
Another perspective of addressing the second question involves reliability of sensor data. In this case, an increasing amount of restrictions can be placed on the radar data that is fused with the navigation data. More specifically, restrictions can be placed on the maximum distance in which a UAV can accurately detect another UAV and therefore plan paths and make decisions accordingly. A starting point is a 100 meter radius, essentially guaranteeing that all UAVs are in sight. From there, the radius can be reduced to 10 meters and finally one meter. Table 5.5 provides the results for each radar distance. Here, it should be noted that EKF and UKF did not include radar data as an input, and thus their results are unaffected by these restrictions.

Ideally, the typical outcome should be that the prediction error increases as radar data becomes more restrictive. However, this only happens in scenario 1 under ANN and in scenario 4 under ANN+DSET. In fact, error actually steadily decreases in scenarios 2 and 5 under ANN. A likely explanation is that the six UAVs remain close together for a majority of the time, with exception of the last 100 seconds of the timeline. Thus, this table is relatively inconclusive in addressing any of the three questions, primarily due to the fact that it averages error over the entire timeline.

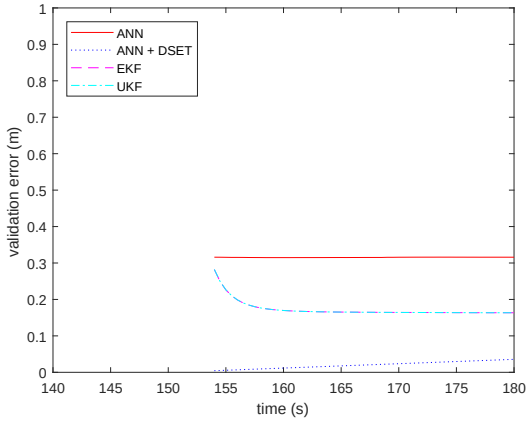
Therefore, a logical next step is to revisit these results with a knowledge of each individual time step.

Fig. 5-8 provides the same results under a 100 meter radar over the course of time. Each subfigure represents each scenario with both methods represented as functions of validation error versus time. In this case, only validation accuracy is visible, given that the first third of each timeline is blank due to the training processes. It should be noted the EKF and UKF give the impression of being one single curve, as their values are extremely close together although not equal. So far, each plot demonstrates that ANN+DSET is most effective for near-term prediction while EKF and UKF is favorable for long term prediction,as evidenced in Scenario 3 and the full timeline. ANN, meanwhile never produces the lowest error at any point in time. These observations help to verify the results given in Table 5.2, in which the ANN, EKF, and UKF models are slightly more reliable than ANN+DSET when fewer and fewer training examples exist. The emphasis on slightly is verified by the fact that only Scenario 3 the full timeline depict an intersection in the any class of methods (meaning that EKF and UKF crossovers with one another do not count), and the Kalman filters do not begin to outperform ANN+DSET until after multiple minutes.

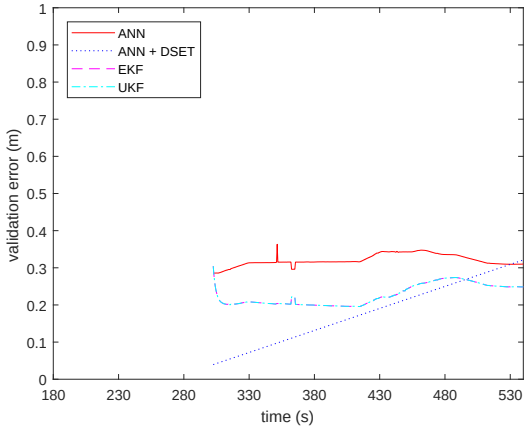
Fig. 5-9 depicts the results of the 10 meter column in a similar format. When comparing each of the subfigures to their 100 meter counterparts, a key observation is that ANN+DSET remains virtually unchanged while ANN undergoes some minor but noticeable changes. As a result of these changes, ANN never outperforms ANN+DSET, although it comes close to crossing over at the end of the full timeline. Essentially, this helps to address the second and third questions. When attempting to extrapolate from UAV data, there appear to be fewer limitations with DSET when observed from an over-time basis, given that ANN+DSET does not change so far while ANN exhibits at least some changes. It is also worth noting that the ANN functions exhibit noticeable bumps over time, presumably due to minor disparities



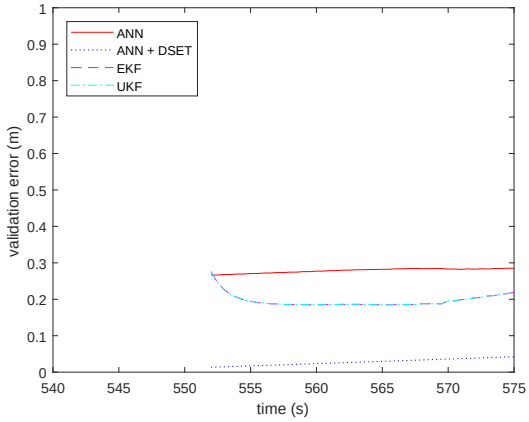
(a) Scenario 1



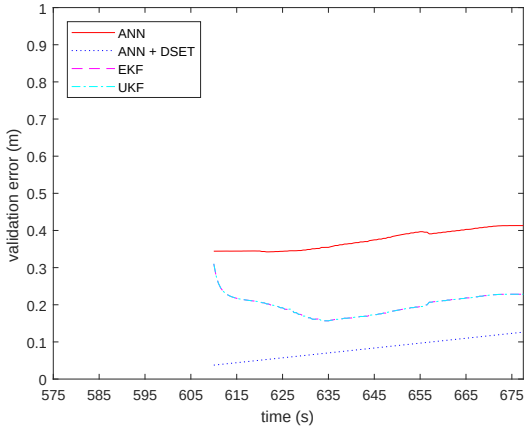
(b) Scenario 2



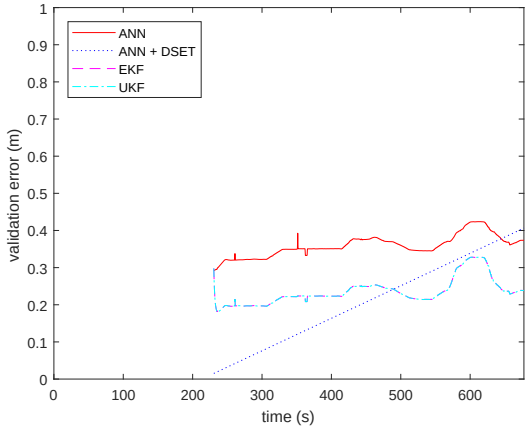
(c) Scenario 3



(d) Scenario 4

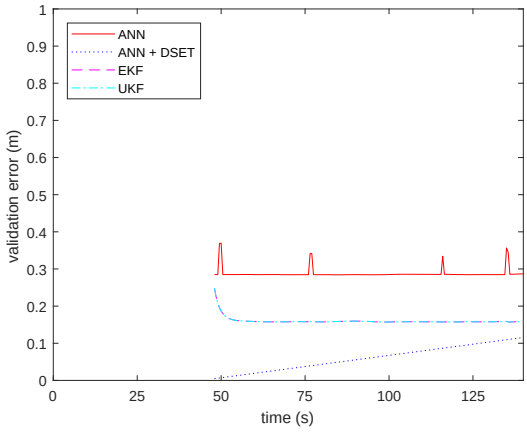


(e) Scenario 5

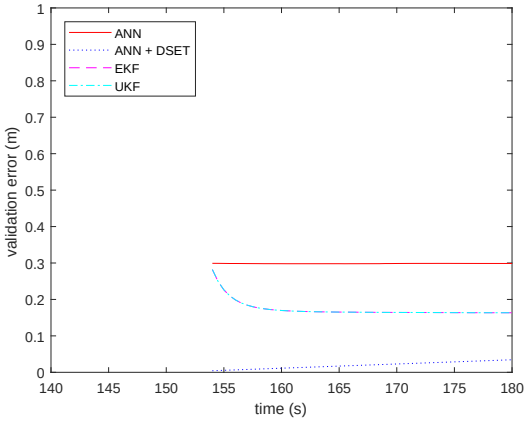


(f) Full timeline

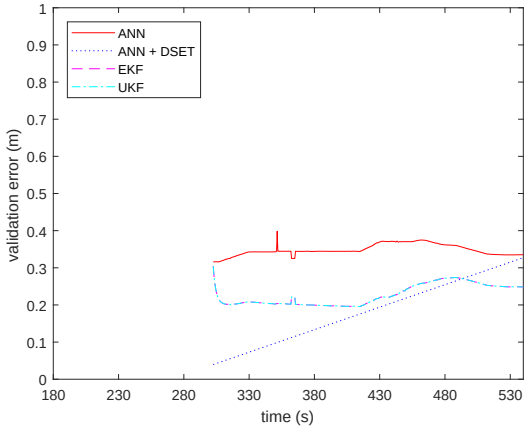
Figure 5-8: Plots of validation accuracy versus time for ANN and DSET/ANN methods under 100 meter radar data.



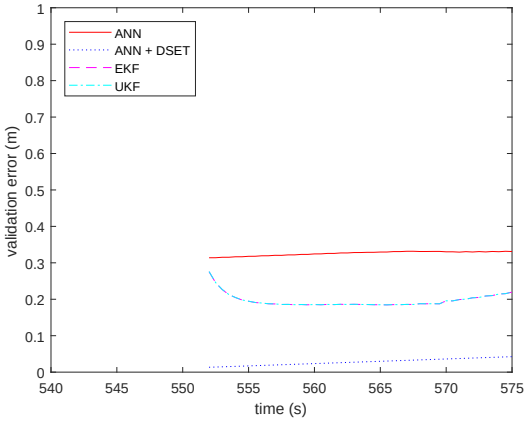
(a) Scenario 1



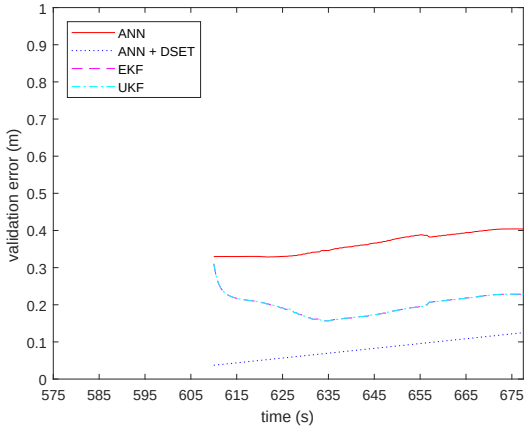
(b) Scenario 2



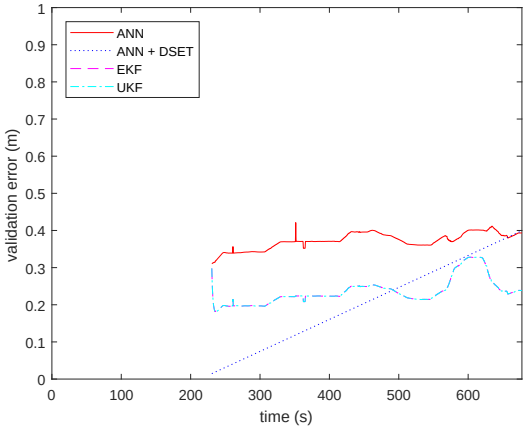
(c) Scenario 3



(d) Scenario 4



(e) Scenario 5



(f) Full timeline

Figure 5-9: Plots of validation accuracy versus time for ANN and DSET/ANN methods under 10 meter radar data.

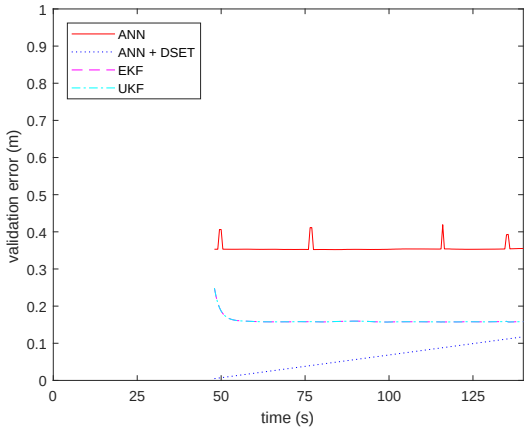
between predictions and results, which also appear to be nonexistent under scenarios 2, 4, and 5.

With the inclusion of DSET, however, error remains stable and does not react to these changes, thereby further ensuring fewer limitations under the ANN+DSET method. When revisiting the results from Table 5.2, however, there are more restrictions when only considering the average error over time. Thus, there is not a single answer to this question from all angles. Instead, it ultimately depends on how to interpret the prediction accuracy. As for the third question, the tighter radar restrictions can be considered an increasingly complex test case. Again, prediction accuracy, particularly near-term prediction, does remain more reliable under ANN+DSET given the method's resistance to change so far.

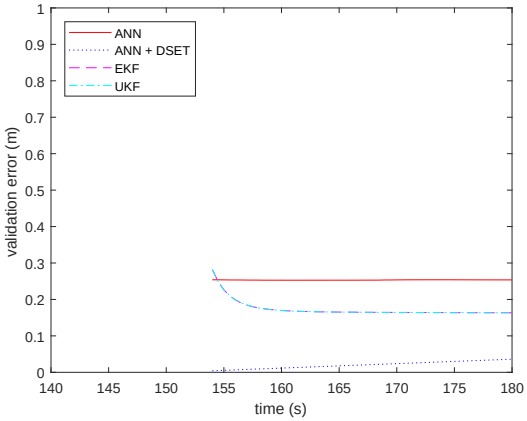
Finally, Fig. 5-10 provides similar results under one meter radar capabilities. Again, ANN+DSET is able to successfully resist change, while ANN exhibits further minor changes. This results in the latter briefly outperforming the former once again toward the end of the full timeline. For this reason, the answers to the second and third question remain largely the same. In addition, the first question can now be clearly answered, as the maximum near-term prediction accuracy (and therefore minimum near-term error) over time is consistently lower with DSET than without.

5.4.2 Multi-UAV Analysis

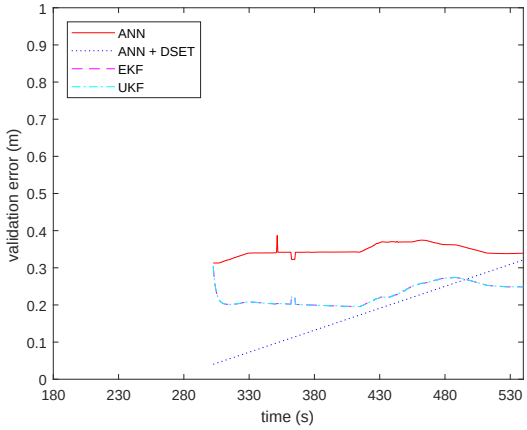
To provide a further dimension of comparison and analysis, this subsection addresses the algorithms from the perspective that inputs and outputs of all six UAVs are evaluated together in one function, rather than evaluating each UAV individually as was done previously. In both cases, however, the results are averaged over all the UAVs for the sake of consistency. Like before, the analysis begins with the average results over time at 25% and 66% validation accuracy, as given in Table 5.6. Again, the non-DSET models are able to handle a decreased number of training samples



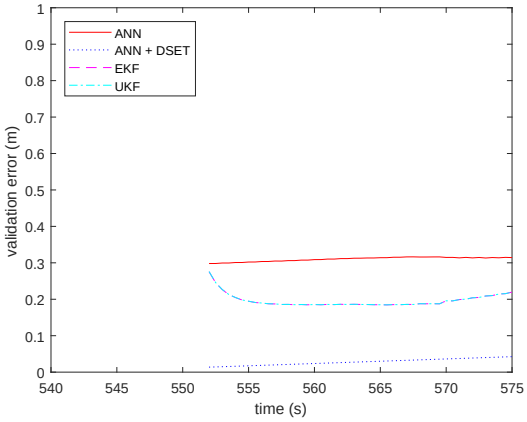
(a) Scenario 1



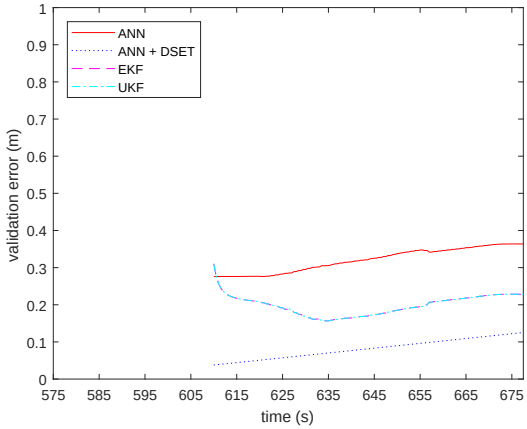
(b) Scenario 2



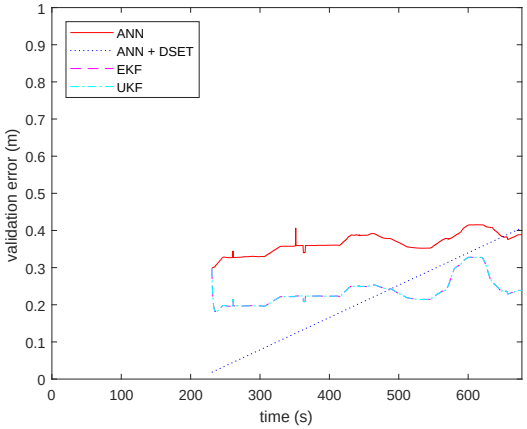
(c) Scenario 3



(d) Scenario 4



(e) Scenario 5



(f) Full timeline

Figure 5-10: Plots of validation accuracy versus time for ANN and DSET/ANN methods under 1 meter radar data.

Table 5.6: Simulation results evaluating average prediction error over time by validation ratio, method, and scenario.

	25% validation				66% validation			
Scenario	ANN	DSET	EKF	UKF	ANN	DSET	EKF	UKF
1	.4320	.2882	.2231	.2231	.4175	.2625	.2085	.2085
2	.4834	.3009	.2327	.2327	.4105	.3213	.2249	.2249
3	.3838	.5322	.2800	.2800	.4500	.5026	.2491	.2491
4	.4359	.6290	.2444	.2444	.3808	.6136	.2997	.3056
5	.4366	.7844	.2563	.2563	.3789	.7586	.2135	.2135
Full	.3988	.4324	.2948	.2948	.3808	.4449	.2427	.2427

Table 5.7: Simulation results evaluating average prediction error over time by radar capability, method, and scenario.

	100 meters				10 meters				1 meter			
Scenario	ANN	DSET	EKF	UKF	ANN	DSET	EKF	UKF	ANN	DSET	EKF	UKF
1	.4175	.2625	.2085	.2085	.4336	.2755	.2085	.2085	.3882	.2748	.2085	.2085
2	.4105	.3213	.2249	.2249	.3596	.2975	.2249	.2249	.3884	.3250	.2249	.2249
3	.4500	.5026	.2491	.2491	.4161	.4839	.2491	.2491	.4180	.4613	.2491	.2491
4	.3808	.6136	.2997	.2997	.4076	.6093	.2997	.2997	.4688	.6401	.2997	.2997
5	.3675	.7535	.2135	.2135	.4077	.7543	.2135	.2135	.3925	.7344	.2135	.2135
Full	.3808	.4449	.2427	.2427	.4059	.4341	.2427	.2427	.4251	.4236	.2427	.2427

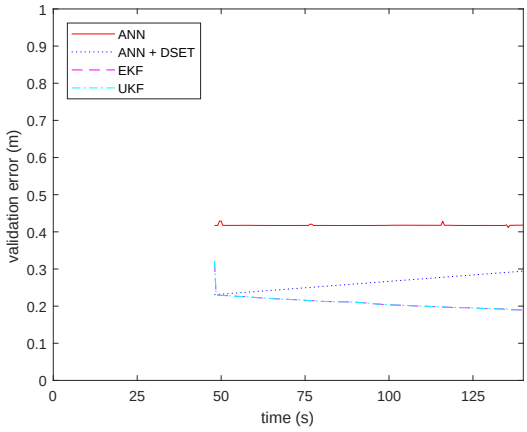
quite well, as the prediction error steadily decreases over every scenario except 3 in ANN and 4 in EKF and UKF. With ANN+DSET, however, scenarios 1, 3, 4, and 5 result in this kind of decrease. Under these observations thus far, it is possible that under multi-UAV analysis, ANN+DSET seems to have a similar amount of, or even fewer limitations, than the other methods, which is contrary to the results in the single-UAV analysis.

Table 5.7 provides the average error over time by radar data, in a manner similar to that of Table 5.5. Under scenarios 4 and the full timeline, ANN follows a more predictable pattern of error increasing as radar data becomes more restrictive. For all other scenarios, there is no noticeable pattern. For ANN+DSET, error consistently

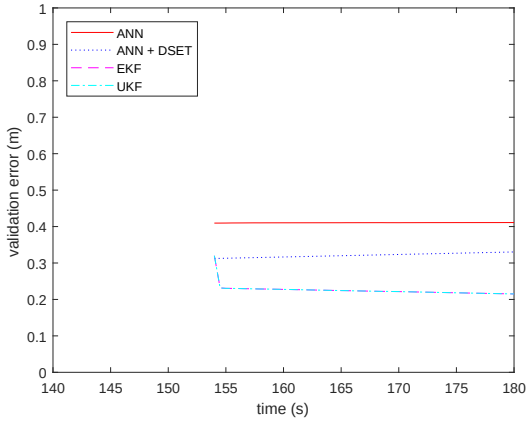
decreases in scenarios 3 and 5 as well as in the full timeline. Thus, there seem to be no similarities between these results and the single-UAV results. This is likely due to the fact that evaluating all UAVs together tends to blur the distinctions of when two UAVs are close to one another (which result in no real change in accuracy when tightening radar data) versus when two UAVs are far apart. Furthermore, if both trends tend to exist among different pairs of UAVs in the same scenario, it is nearly impossible to evaluate average error over time in terms of radar reliability. Thus, it is again worth looking into individual error amounts over time.

Fig. 5-11 depicts the prediction error over time for 100 meter radar data in a manner similar to that of Fig. 5-8. One of the most significant changes between these two sets of results is the fact that DSET+ANN no longer outperforms the other methods except at some initial points in scenarios 1 and 2 as well as in the full timeline. Essentially, in all timelines, ANN+DSET error has encountered a sharper increase than ANN, which has resulted in less disparity between the two error functions as well as the ANN's superior performance in 4 and 5. EKF and UKF, meanwhile, outperform both of the other methods at nearly every point.

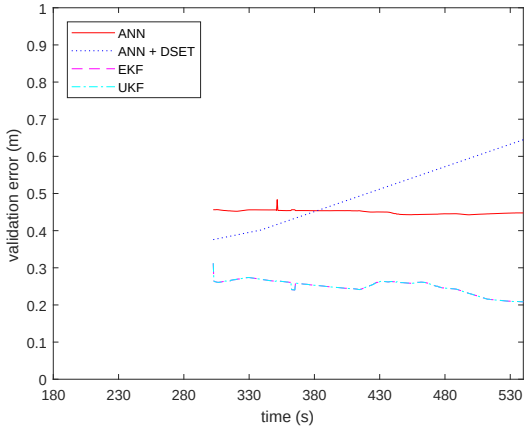
Before moving on to the other radar settings, the three questions can already be revisited regarding this new basis of comparison. For instance, in multi-UAV analysis the maximum near-term prediction accuracy with DSET does not always exceed that without, although it still holds true in the majority of the timelines. When extrapolating from UAV data, there are now mixed results on whether there are more or fewer limitations with DSET. ANN+DSET is limited by some minor disparities between predictions and outcomes, as indicated by the small jumps in some of the corresponding plots, and it is limited somewhat by restrictions in radar data for near-term prediction. However, ANN+DSET is significantly limited in how well it can evaluate multiple UAVs at a time when compared to single-UAV analysis and is somewhat limited by average prediction over time when fewer training samples



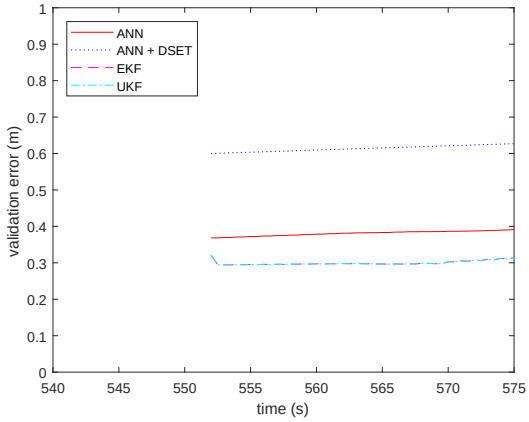
(a) Scenario 1



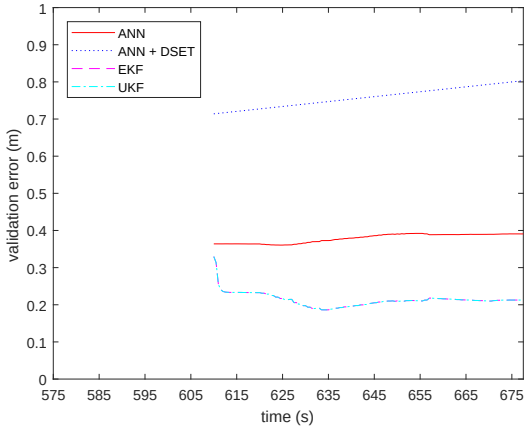
(b) Scenario 2



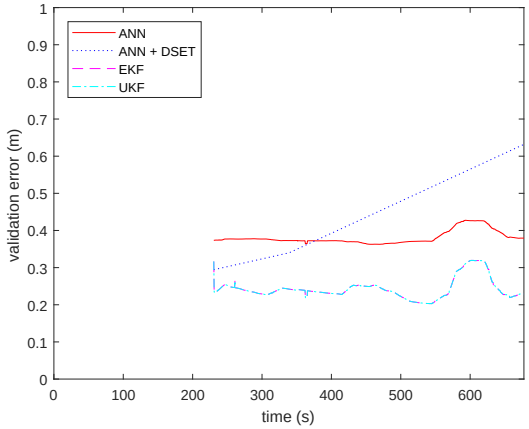
(c) Scenario 3



(d) Scenario 4



(e) Scenario 5



(f) Full timeline

Figure 5-11: Plots of validation accuracy versus time for ANN and DSET/ANN methods under 100 meter radar data.

and more validation samples are available.

As can be expected, Fig. 5-12 presents the prediction error for the 10 meter radar capability. Like in single-UAV analysis, ANN+DSET remains virtually unchanged, while ANN encounters slight changes. EKF and UKF are entirely unchanged, as they are unaffected by radar data. As a result, crossovers occur later in scenario 3 and the full timeline. A new observation, however, is that the jumps in ANN error are much more pronounced than in the previous figure. While Figs. 5-8 and 5-9 also contain differences in jumps, they were much more minor compared to the changes occurring between Figs. 5-11 and 5-12. Fig. 5-13 continues these trends but with less prominent jumps in ANN error.

5.5 Discussion

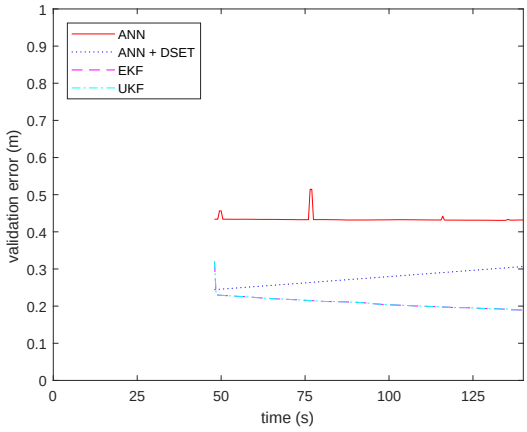
To provide meaningful discussion on such a wide array of results, it is beneficial to revisit the three questions introduced in the previous section:

1. Under which conditions is the maximum near-term prediction accuracy achieved?

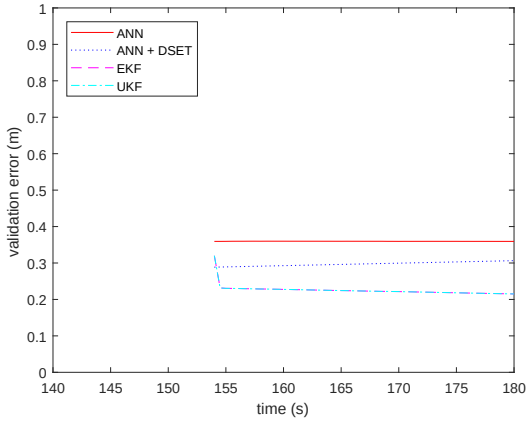
2. Under which conditions is extrapolation from UAV data more limited with DSET than without?

3. Under which conditions does near-term prediction accuracy remain more reliable with DSET (i.e. less prone to significant change) than without as test cases become increasingly complex?

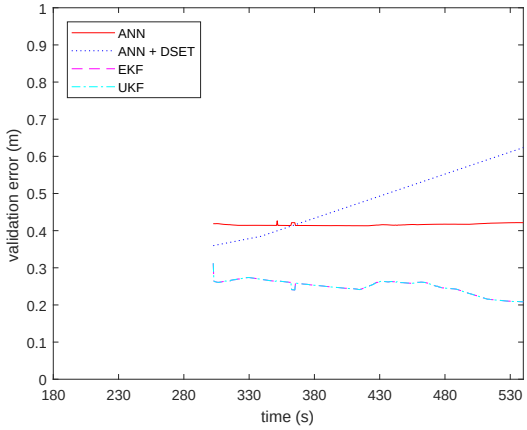
For the first question, when evaluating one UAV at a time, the minimum value of the ANN+DSET error function always exceeded that of the other methods, as clearly given in the corresponding figures. In multi-UAV analysis, this condition did not hold true when compared to EKF and UKF, but when compared to ANN alone, despite a few exceptions to this, but the condition mostly still held true for most of the



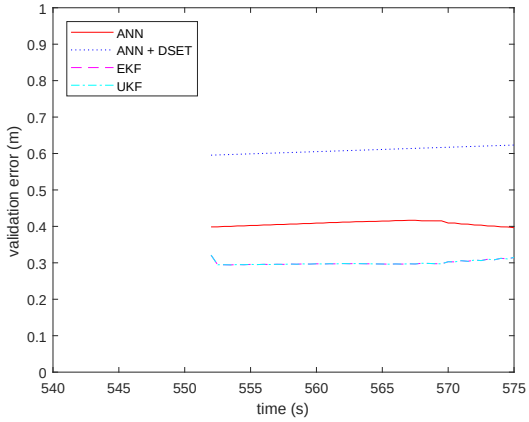
(a) Scenario 1



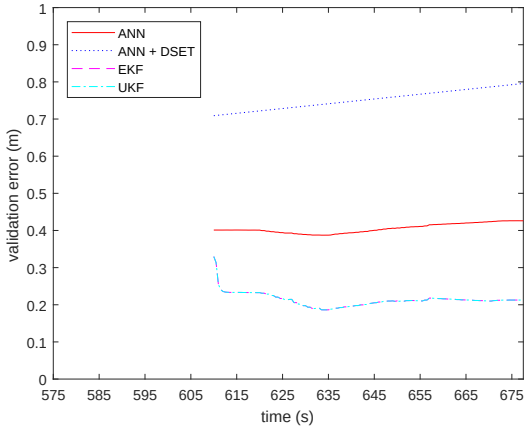
(b) Scenario 2



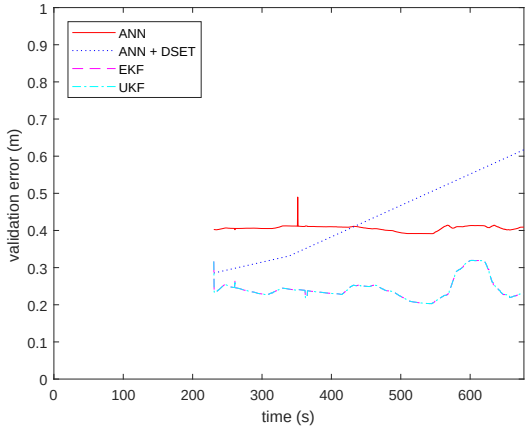
(c) Scenario 3



(d) Scenario 4

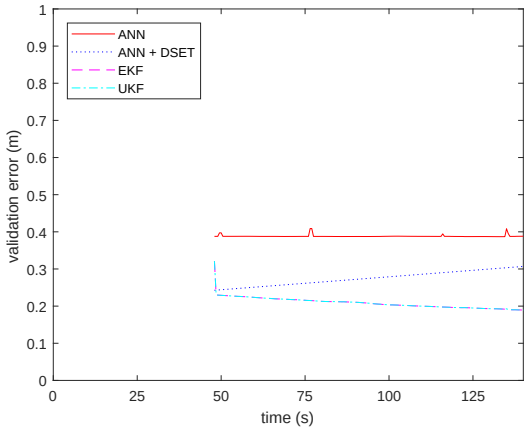


(e) Scenario 5

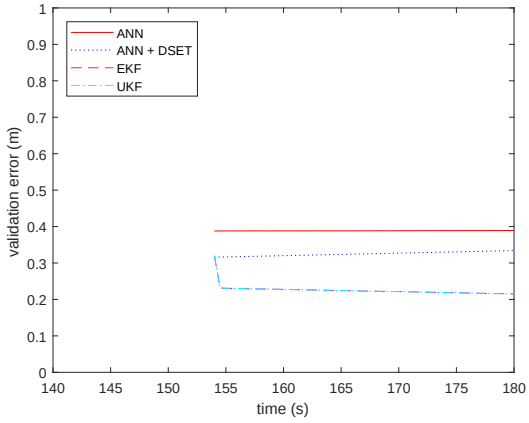


(f) Full timeline

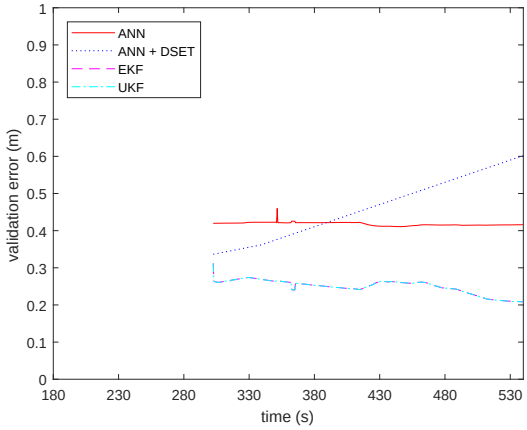
Figure 5-12: Plots of validation accuracy versus time for ANN and DSET/ANN methods under 10 meter radar data.



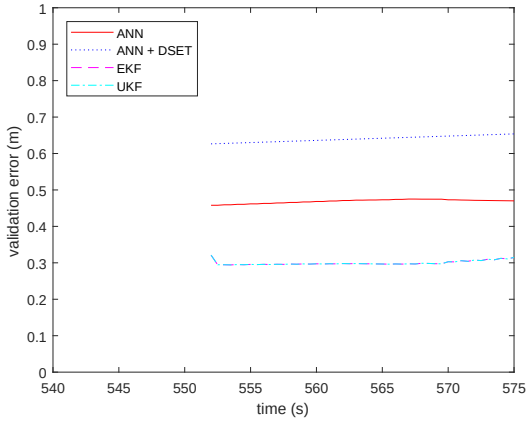
(a) Scenario 1



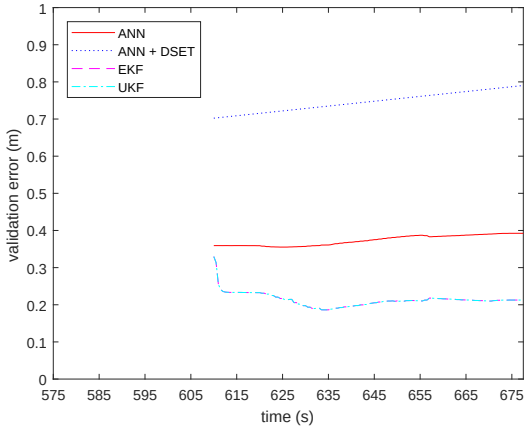
(b) Scenario 2



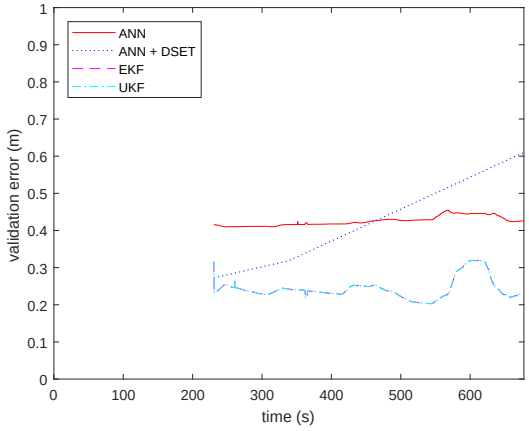
(c) Scenario 3



(d) Scenario 4



(e) Scenario 5



(f) Full timeline

Figure 5-13: Plots of validation accuracy versus time for ANN and DSET/ANN methods under 1 meter radar data.

given timelines. Ultimately, when seeking the highest possible prediction accuracy for a UAV, more often than not, the inclusion of sensor fusion via DSET is a highly valuable asset.

As for the second question, the consensus is less clear, as all of the possible limitations that have been integrated into the experiment must be taken into consideration. For validation accuracy, in which there are progressively fewer training samples with which to work and increasingly more validation samples for which to account, there appears to be more limitations with DSET than there are without. In terms of increasingly restrictive sensor data, such as the radar data in this case, DSET seems to be less limited as evidenced the lack of change in DSET error when radar data tightens, compared to modest changes in ANN error under the same circumstances. Comparisons to EKF and UKF are inconclusive, as these methods were implemented without radar data as an input.

When adjusting for less predictable UAV behavior, ANN+DSET is better able to provide a steady amount of accuracy when compared to ANN, which experienced various jumps in error under certain circumstances, thus accounting for another limitation in which DSET is immune. When switching from single-UAV analysis to multi-UAV analysis, however, the inclusion of multiple UAVs serves as the most significant limitation to DSET. Thus, conclusions can be made that overall there are slightly more limitations with DSET, but in single-UAV prediction, there are much fewer, and in multi-UAV prediction, there are much more.

The third question bears some similarity to the second, but here the primary concern is the amount of change in each case. Again, it is best to answer this question piece by piece. When discussing the complexity of test cases, the reduction of training samples, increases in validation samples, and increased numbers of inputs and outputs are valid bases for observation. In terms of training and validation, because only the average prediction accuracy was evaluated when adjusting validation ratios, it

is helpful to observe the different timelines, since some have more samples of both training and validation than others. In single-UAV analysis, Figs. 5-8 through 5-10 show that as the length of the timeline increases, ANN+DSET error tends to change more rapidly, ranging from zero meters to 0.4 meters.

Meanwhile, ANN error mostly hovers around 0.3 to 0.4 meters, resulting in less reliability with DSET than without. This mostly holds true in multi-UAV analysis as well, as evidenced by Figs. 5-11 through 5-13. The only notable difference is that ANN jumps are more prominent, resulting in more significant change, but this only occurs under fewer samples. Thus, in terms of number of samples, the answer to the third question so far is no. For the next factor of complexity, increased inputs and outputs, ANN+DSET changes very rapidly overall in multi-UAV analysis, although ANN is more prone to sudden jumps in error. Overall, however, the inclusion of DSET is less reliable when test cases become increasingly complex.

In answering all three questions, conclusions can be drawn that DSET is mostly successful in one area of analysis, partly successful in another, and mostly unsuccessful in a third area. Now, the next step is to analyze what all of this means in the overall context of UAV data prediction and the viability of sensor fusion. In general, the results have shown a considerable amount of success in enhancing prediction accuracy with sensor fusion by means of the innovative DSET component. The only catch is that the context and the conditions of DSET's success matters greatly. When evaluating a single UAV, highly effective near-term predictions can be made when fusing sensory data such as navigation and radar. When making longer term predictions or evaluating multiple UAVs at a time, this implementation of sensor fusion bears some limitations. However, it is also merely scratching the surface on the overall capabilities of data fusion. Thus, many future opportunities still exist.

5.6 Conclusion

This chapter presented four machine learning mechanisms for predicting an unmanned aerial vehicle's location and trajectory to aid in an increased understanding of UAV behavior and help to bridge the gaps between expectations and outcomes. The first approach was an artificial neural network capable of taking multiple inputs from one or more UAVs, such as time steps and radar data, outputting navigational coordinates, and training past inputs and outputs. The second approach was based on an extended Kalman filter (EKF) and some UAV-based mathematical modeling given by [208]. The inputs and outputs were based solely on navigational coordinates and headings. The third approach, meanwhile, was based on an unscented Kalman filter (UKF) and followed a closely similar approach to the EKF method. Finally, the fourth approach combines a neural network with Dempster-Shafer Evidence Theory. This approach takes the same inputs but for the first time uses DSET to more accurately determine latitudinal and longitudinal coordinates, while a simpler ANN predicts the altitude and heading. The results ultimately indicate that the ANN+DSET approach produces significantly higher average accuracy than the ordinary ANN method as well as the lowest near-term prediction accuracy over time. It also outperformed EKF and UKF a vast majority of the time when evaluating one UAV at a time. The ANN, EKF, and UKF approaches, however, succeed more in long term prediction and in evaluating multiple UAVs at a time.

Chapter 6

UAV Flight Performance through Adaptive Trajectory Prediction

Much like in Chapter 4, the next step after comparing and evaluating different methods (in this case, a choice of ANN, ANN+DSET, EKF, or UKF) is to develop a system that dynamically selects a technique based on which one has, or is destined to have, the lowest prediction error at a given point in time. This is done by checking the expected performances of the four different candidate methods and bringing the dominant choice into the foreground of data prediction. The technique was weighed against the same dataset described in Chapter 5 and evaluated across a similarly extensive set of experimental parameters. Overall, the results indicate a significant amount of cases in which the proposed algorithm outperforms the individual methods.

6.1 Introduction

As was the case in Chapters 3 and 4, one of the most prominent paths to ensuring safe aviation is being able to accurately predict some numerical quantity related to a crucial aspect of the flight. In the case of manned flight safety, that is the level of cognitive workload, which can correlate with different mentally intensive activities. In

the case of unmanned flight, cognitive workload still has a role in scenarios in which significant amounts of control come from remote human pilots. The future of UAV applications, however, lies within the increasing role of autonomy. Hence, accurate prediction of sensory data from the UAV itself becomes of greater importance in the role of unmanned flight safety.

Chapter 5 presented a start to this approach by improving an artificial neural network with the inclusion of data fusion. This new method was compared against a standard ANN as well as other algorithms relevant to UAV data prediction and showed promising results in many, but not all, situations. However, when developing real time applications and transitioning algorithms from theory and simulation to practice, simply comparing methods is not enough. A more useful approach is to take every tool available at one's disposal and apply different approaches at different times depending on which one is perceived to be most effective. This has been demonstrated already in Chapter 4. Hence, a similar approach is given here.

Thus, this chapter aims to utilize the four candidate approaches used in the previous chapter for UAV data prediction. This includes the proposed ANN+DSET technique, the standard ANN, extended Kalman filter, and unscented Kalman filter. From there, the goal is to dynamically switch among the four over each of the UAV timelines (as detailed in Table 5.1) based on which method produces the best results, or is expected to produce the best, at any given point in time.

6.2 Methods and Materials

In developing this algorithm, the key focus was on examining accuracy for each candidate method at each time step and selecting the method with the best accuracy to remain or become the foreground method, while all others are the background methods. A detailed flowchart of this process is presented in Fig. 6-1. From there, the

Figure 6-1: Detailed switching algorithm.

next step is to develop the specific algorithmic procedure, which is given in Fig. 6-2. The experiment was again conducted in MATLAB under the same default experimental parameters given in the beginning of Subsection 5.3.3. In a manner similar to that of the previous chapter, the experiment was conducted across six of the seven dimensions of parameters. When omitting the first variable of method, these are:

1. Time
2. Scenario (as detailed in Table 5.1)
3. Individual UAV and all UAVs combined

```
train_in, val_in, train_out, val_out ← all possible input and output data
output ← all possible accuracy and coordinate data
for all trials do
  for all time partitions do
    for all validation ratios do
      for all radar data restrictions do
        for individual UAVs versus all UAVs combined do
          train first ANN using x, y, h, and θ in outputs
          train second ANN using only h and theta in outputs
          for all training time steps do
            apply EKF filter
            apply UKF filter
          end for
          for all validation time steps do
            ann_out_1 ← predicted x, y, h, and θ of first ANN
            ann_out_2 ← predicted h and θ of second ANN
            for all UAVs (1 if evaulating all UAVs together) do
              dset_out ← predicted x and y using DSET
              store all coordinate data in output
            end for
            ekf_out_1 ← predicted x, y, h, and θ
            ukf_out_1 ← predicted x, y, h, and θ
            get validation error per time step of selected method
            if error is higher than that of another method then
              select method with highest accuracy
            end if
            store error data in output
          end for
        end for
      end for
    end for
  end for
end for
average output over all trials
return output
```

Figure 6-2: Pseudocode of complete experimental process.

4. Radar restrictions (UAVs can be detected within 100, 10, and 1 meters)

5. Percent validation (25 and 66 %)

6. Trial number

From here, the end result is a single MATLAB variable in the form of a cell array. Each cell corresponded to each scenario and consisted of a six-dimensional matrix. The dimensions were time, radar restrictions, data type, UAV number, one UAV versus all, and trial number, respectively. The experiment completed by averaging each matrix across all trials, thereby resulting in a cell array of five-dimensional matrices afterwards.

6.2.1 Evaluation Metrics

As is the case in the previous chapter, prediction error the form of root mean squared error of the latitudinal coordinates, longitudinal coordinates, and altitude. Thus, it is still defined as

$$E = \frac{1}{n}\sum_{i=1}^{m}\sqrt{(x_{i,act} - x_{i,pred})^2 + (y_{i,act} - y_{i,pred})^2 + (h_{i,act} - h_{i,pred})^2} \tag{6.1}$$

when averaging the error across all the validation samples.

6.3 Results and Discussion

While Chapter 5 presented results around three overarching questions, these results simply lie within the goal of having the minimum amount of prediction error possible. From most other aspects, however, presentation of results bears similarity to the previous chapter. This includes tables of error results by scenario and by factors

Table 6.1: Simulation results evaluating average prediction error over time by validation ratio, method, and scenario.

	25% validation			66% validation		
Scenario	100 m	10 m	1 m	100 m	10 m	1 m
1	.0290	.0294	.0292	.0598	.0622	.0628
2	.0125	.0120	.0124	.0204	.0199	.0207
3	.1205	.1215	.1198	.1806	.1812	.1868
4	.0294	.0297	.0299	.0425	.0431	.0432
5	.1071	.1079	.1063	.1437	.1411	.1429
Full	.1486	.1472	.1491	.1991	.1961	.2044

such as percent validation, individual UAV, and radar restrictions. Also included are plots of validation error over time but this time for only one method.

6.3.1 Single-UAV Analysis

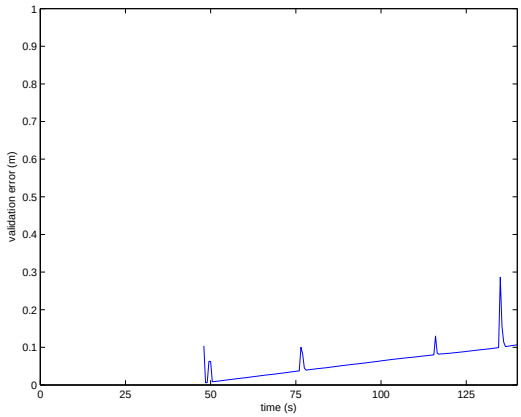
Table 6.1 provides the simulation results for each scenario under 25% and 66% validation for the same three sets of radar data. When observing any variability between restrictions in radar data, there does not appear to be any significant correlation. This is due to the algorithm's ability to switch methods at a time step's notice, thereby avoiding any increase in error that one method may have over another. When adjusting for change in percent validation, there is still a steady increase in error as percent of training data decreases, which differs from results in the previous chapter in which some error values increased while others decreased. Another noticeable change is that there is an almost direct correlation between time length of partition and prediction error, which are two variables that seemed largely uncorrelated previously. Overall, however, the most noticeable change is the significantly lower overall error values in all circumstances when compared to their Chapter 5 counterparts. This observation alone makes the algorithm a success so far.

Table 6.2: Simulation results evaluating average prediction error over time for each individual UAV by method and by scenario.

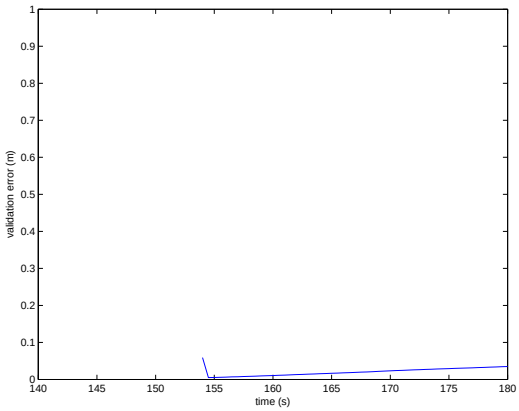
Scenario	UAV 1	UAV 2	UAV 3	UAV 4	UAV 5	UAV 6
1	.0656	.0725	.0610	.0595	.0654	.0351
2	.0254	.0210	.0209	.0212	.0204	.0137
3	.1916	.1720	.2140	.2531	.1765	.0762
4	.0276	.0169	.0173	.0179	.0146	.1610
5	.1559	.0882	.0481	.1194	.1401	.3109
Full	.2098	.1756	.2352	.2157	.2207	.1375

Another perspective of observing results lies within Table 6.2, in which results are compared both by scenario and by individual UAV. When comparing these results with those of Tables 5.3 and 5.4, the proposed algorithm consistently outperforms most of the individual methods. Surprisingly, however, it does not always outperform the ANN+DSET algorithm, in which there are mixed results on which method produces the best accuracy. This is likely due to the fact that the first step often does not invoke the right technique right away as well as the fact that there may be one time step delay when a crossing point occurs. Either or both of these drawbacks are prominent contributing factors to the proposed method's occasional underperformance to ANN+DSET. This deficiency, however, only appears prominent when evaluating individual UAVs. When averaging UAV results together, the latter algorithm maintains supremacy.

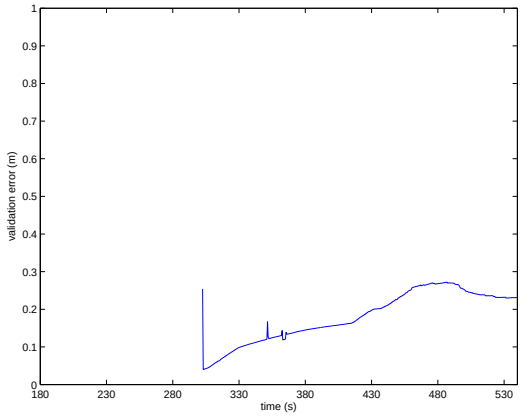
The next three figures depict validation error as a function of time, similarly to the previous chapter. The key difference here, however, is that only one method is evaluated. Fig. 6-3 provides the result in the event of no radar restrictions. Predictably, the function follows similar patterns to those of Fig. 5-8 but for only the best performing algorithms. For instance, in most of the subfigures, ANN+DSET performs best under the entire two timelines. Hence, the respective subfigures in Fig. 6-3 follow the



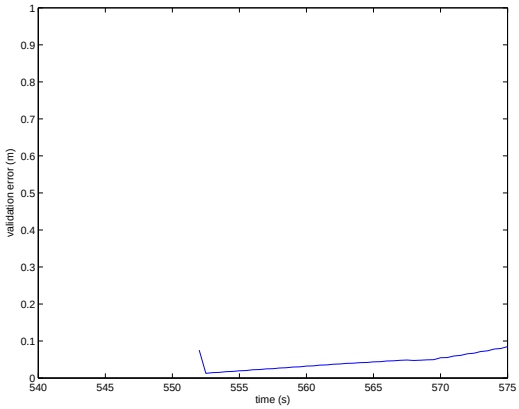
(a) Scenario 1



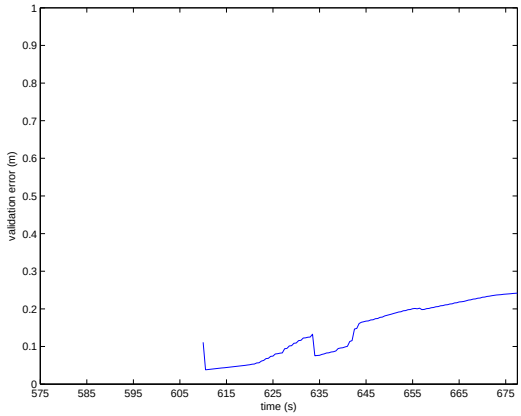
(b) Scenario 2



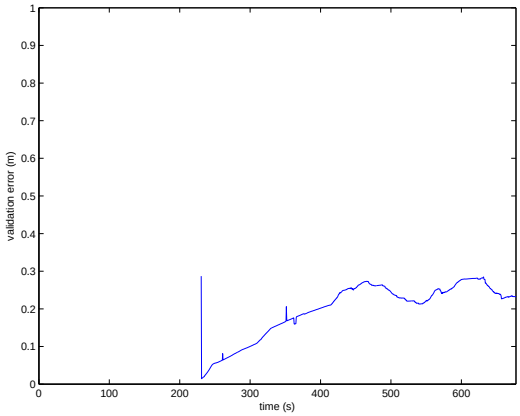
(c) Scenario 3



(d) Scenario 4

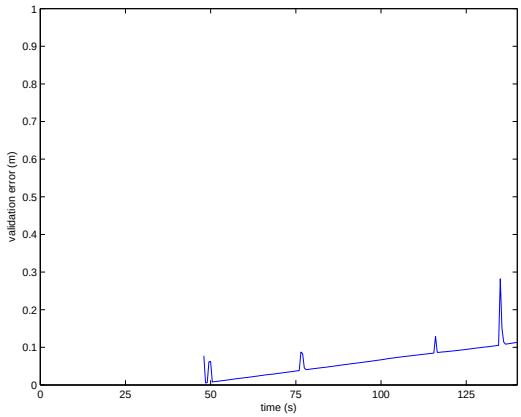


(e) Scenario 5

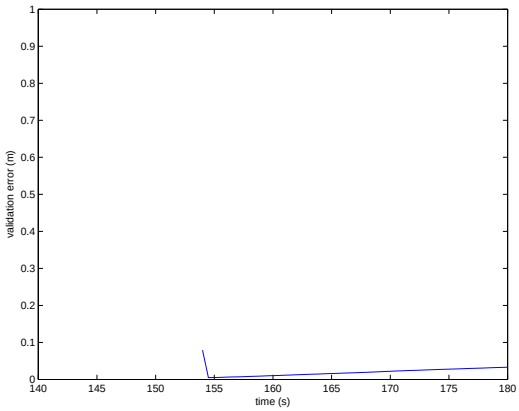


(f) Full timeline

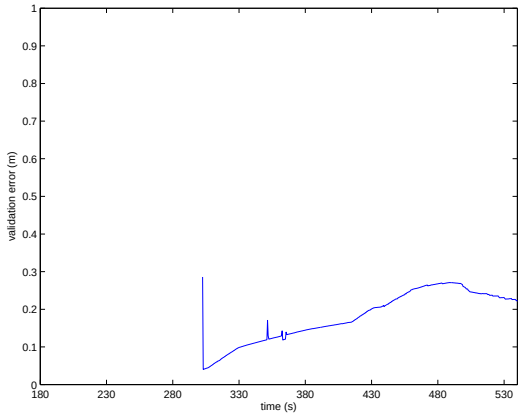
Figure 6-3: Plots of validation accuracy versus time for ANN and DSET/ANN methods under 100 meter radar data.



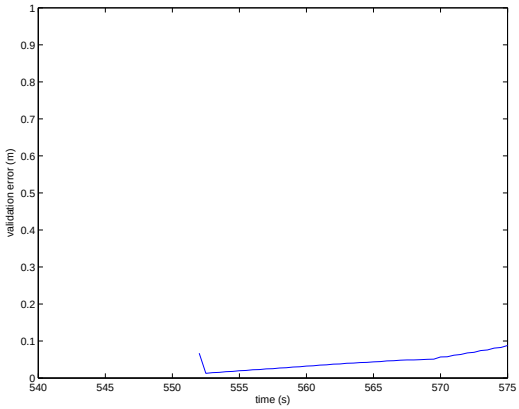
(a) Scenario 1



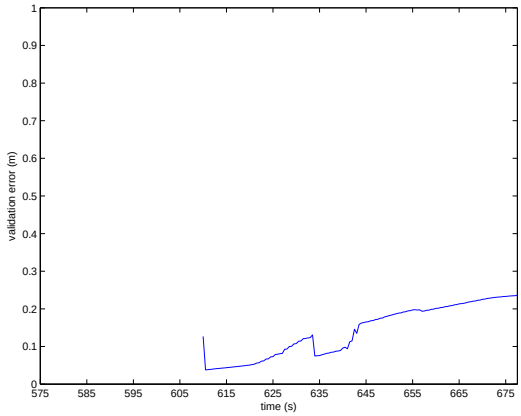
(b) Scenario 2



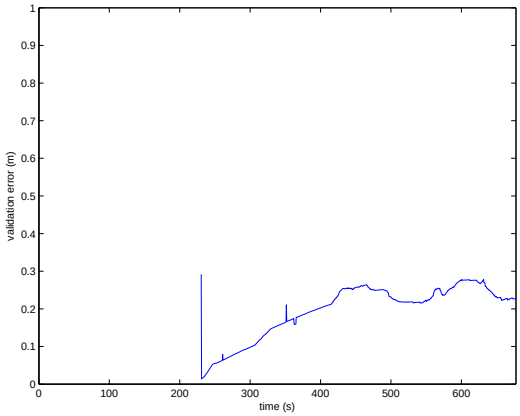
(c) Scenario 3



(d) Scenario 4

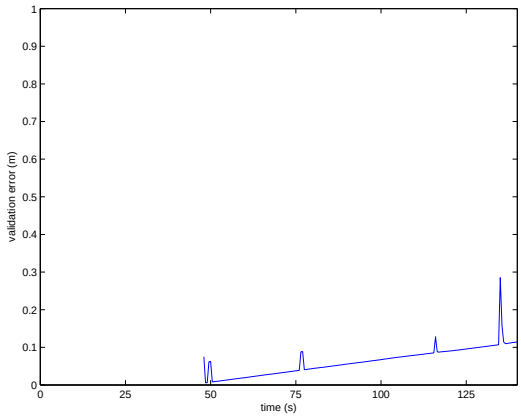


(e) Scenario 5

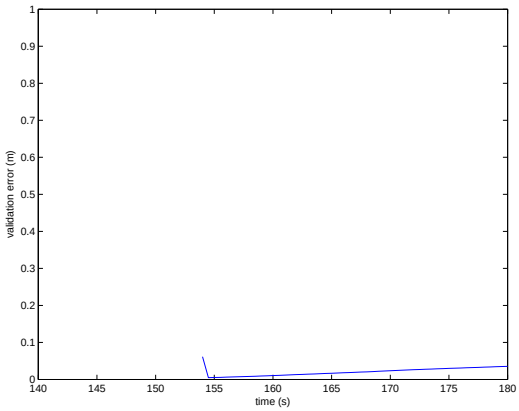


(f) Full timeline

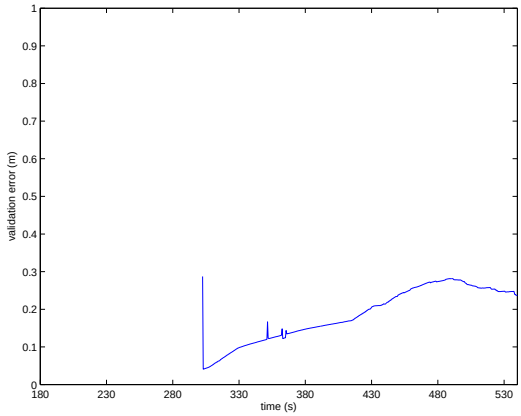
Figure 6-4: Plots of validation accuracy versus time for ANN and DSET/ANN methods under 10 meter radar data.



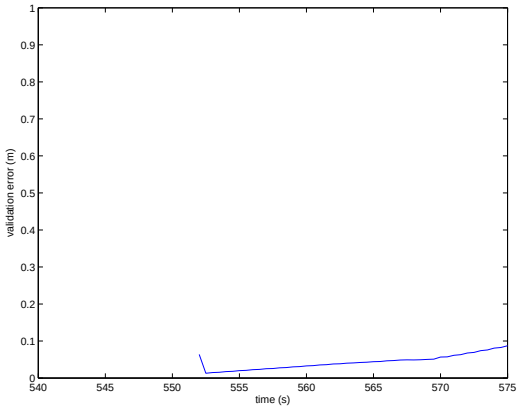
(a) Scenario 1



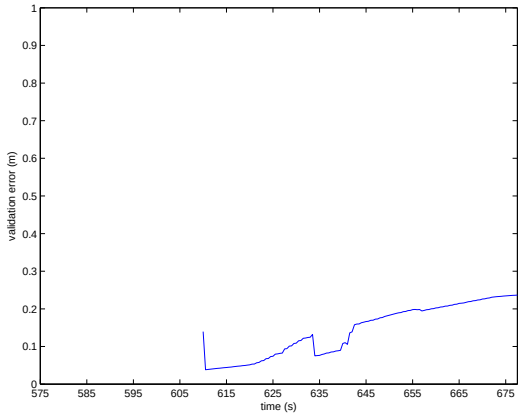
(b) Scenario 2



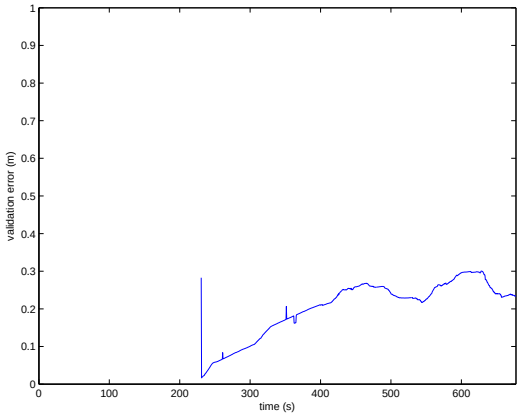
(c) Scenario 3



(d) Scenario 4



(e) Scenario 5



(f) Full timeline

Figure 6-5: Plots of validation accuracy versus time for ANN and DSET/ANN methods under 1 meter radar data.

Table 6.3: Simulation results evaluating average prediction error over time by validation ratio, method, and scenario.

	25% validation			66% validation		
Scenario	100 m	10 m	1 m	100 m	10 m	1 m
1	.2227	.2224	.2229	.2040	.2064	.2078
2	.2336	.2344	.2342	.2255	.2253	.2256
3	.2794	.2787	.2684	.2493	.2492	.2493
4	.2612	.2611	.2609	.3056	.3010	.3062
5	.2563	.2563	.2563	.2162	.2163	.2160
Full	.2911	.2947	.2922	.2428	.2423	.2427

same pattern minus some minor jumps. In the remaining subfigures, however, EKF and UKF perform best towards the ends of the timelines. Thus, the patterns reflect all three selected methods. Figs. 6-4 and 6-5 provide equivalent results under the 10 meter and 1 meter radar restrictions, respectively. Minus a few minor alterations in the curves, these graphs bear the same conclusions as those in the previous figure. It is worth noting that all of these plots involve relatively large starting error values followed by immediate downward jumps into the aforementioned lines and curves. This is also due to the fact that the optimal technique is not always selected at the start of the algorithm. Some differences are barely noticeable, such as in Subfig. 6-4d, while others are highly significant, as evidenced by Subfig. 6-4f. Overall, the results show drastic performance improvement for the proposed algorithm under all timelines in single-UAV analysis.

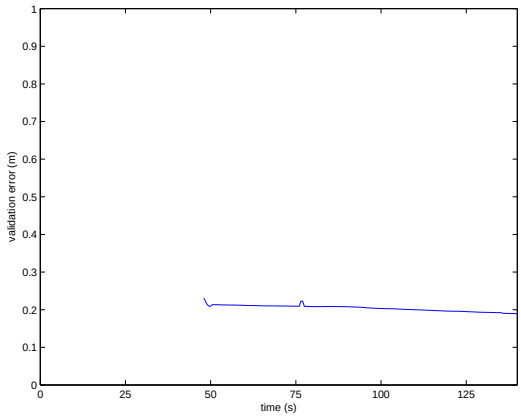
6.3.2 Multi-UAV Analysis

Table 6.3 presents a table of error values in a similar form to that of Table 6.1, in which results can be compared by timeline, radar restriction, and percent validation. This time, however, the results are obtained when taking the inputs and outputs of all

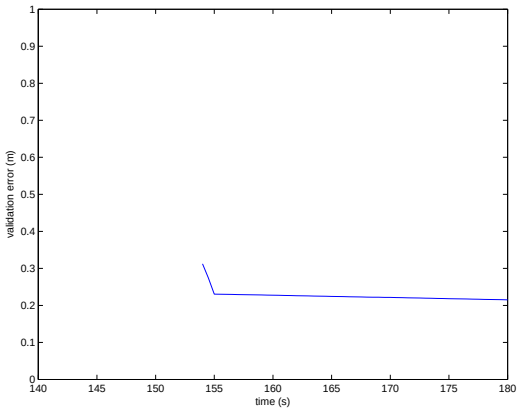
UAVs combined rather than evaluating one UAV at a time and averaging the results together. Like before, noticeable correlation among radar restrictions. A key difference, however, is that in the majority of timelines, error now consistently decreases when percent validation increases. In addition, there is less of a noticeable pattern between error and length of timeline, as was the case in the previous subsection. The full timeline, however, does consistently result in the highest error under 25% validation and the third highest under 66%.

When comparing the radar restricted multi-UAV data to that of the previous chapter, the highest performing individual methods (in this case, EKF and UKF) actually produce slightly better accuracy than the adaptive method in some cases, while the reverse is true in others. In all of these cases, they are by negligible margins, thus indicating that the proposed method performs neither better nor worse overall when considering only multi-UAV data at 66% validation under different radar restrictions. When considering 25% validation, however, the adaptive method produces the best average accuracy in five out of six timelines, even when radar restrictions under the proposed method are greater than those of individual methods.

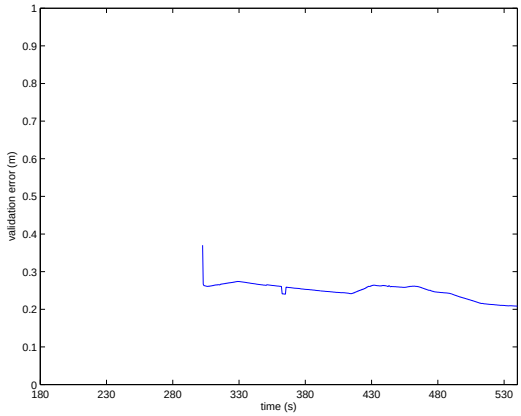
The plots of multi-UAV validation error over time for each timeline are presented in Fig. 6-6. In this case, the best performing multi-UAV methods in Chapter 5 were EKF and UKF. Thus the curves given in this figure closely follow the two methods. Another key observation is that the drop in error after the initial time step is significantly steeper in many cases, as most clearly evidenced in Subfig. 6-8e, while other respective drops are barely noticeable at all, such as in Subfig. 6-6a. These same trends continue in Figs. 6-7 and 6-8. Because there was no significant crossover point in the respective Chapter 5 graphs, the differences in error here are relatively negligible. From the strict perspective of dynamically selecting the best method for highest accuracy, however, this algorithm is still a success. It is simply less prominent than it is in the previous subsection.



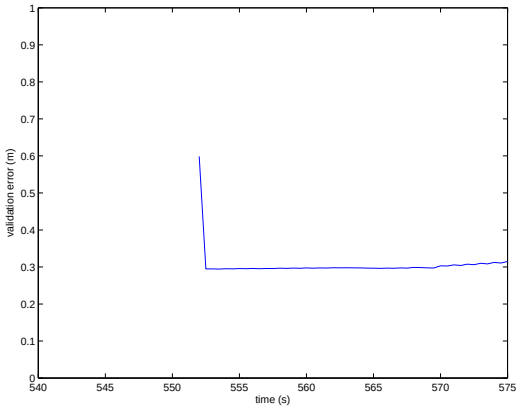
(a) Scenario 1



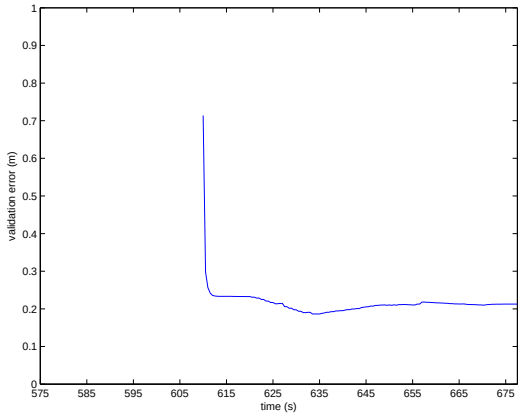
(b) Scenario 2



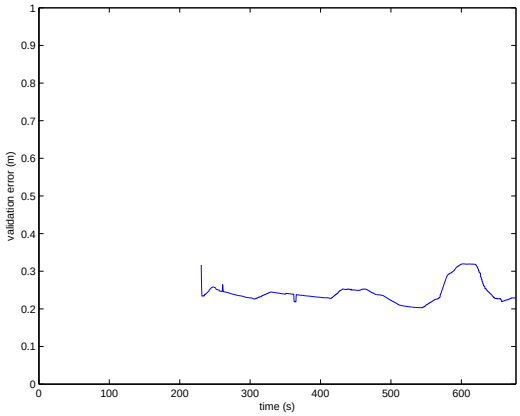
(c) Scenario 3



(d) Scenario 4

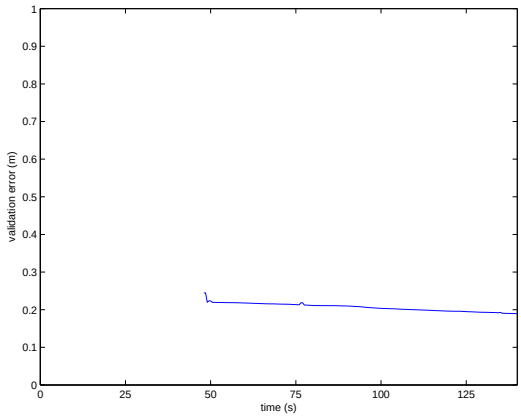


(e) Scenario 5

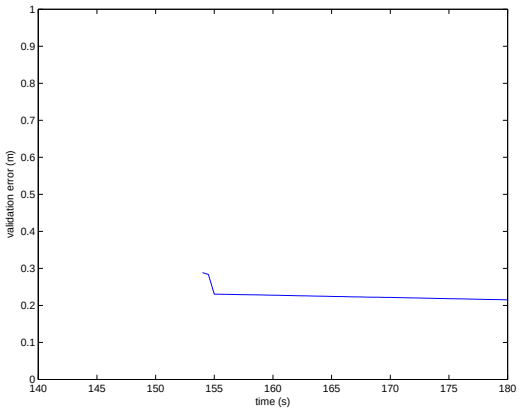


(f) Full timeline

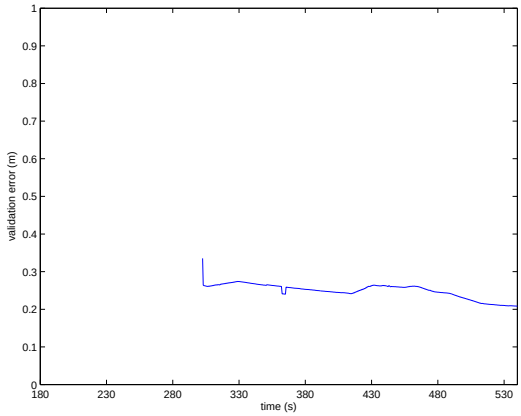
Figure 6-6: Plots of validation accuracy versus time for ANN and DSET/ANN methods under 100 meter radar data.



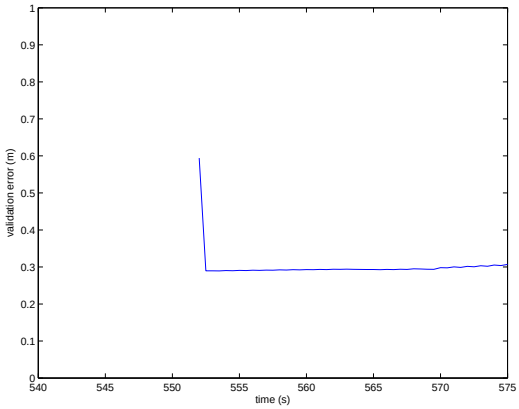
(a) Scenario 1



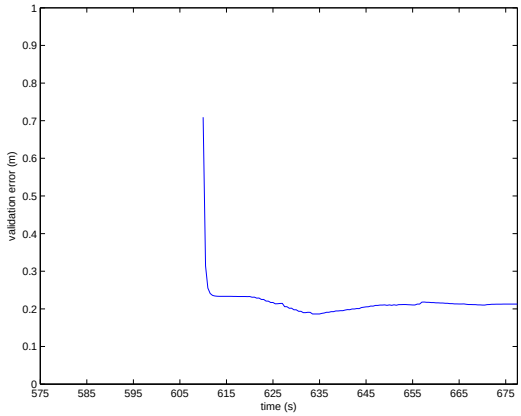
(b) Scenario 2



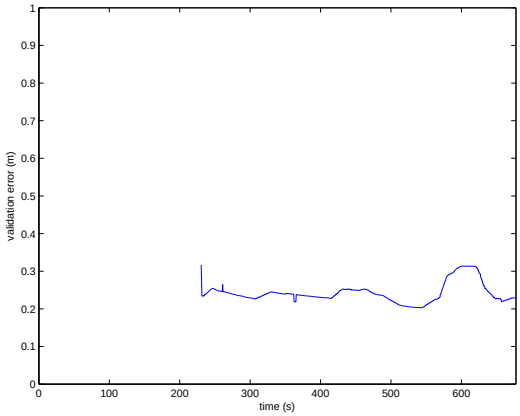
(c) Scenario 3



(d) Scenario 4

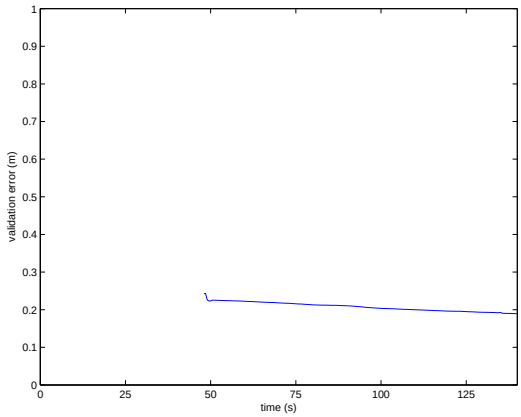


(e) Scenario 5

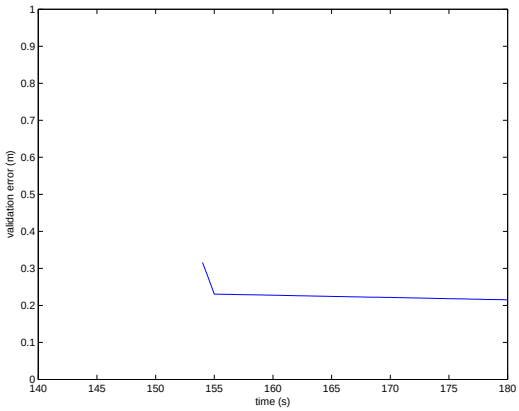


(f) Full timeline

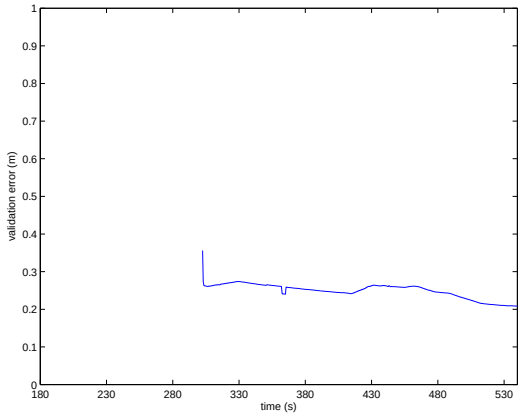
Figure 6-7: Plots of validation accuracy versus time for ANN and DSET/ANN methods under 10 meter radar data.



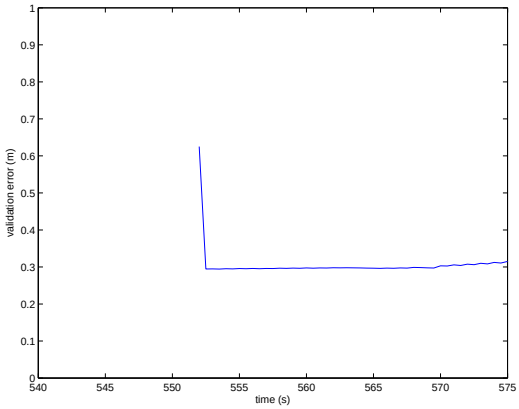
(a) Scenario 1



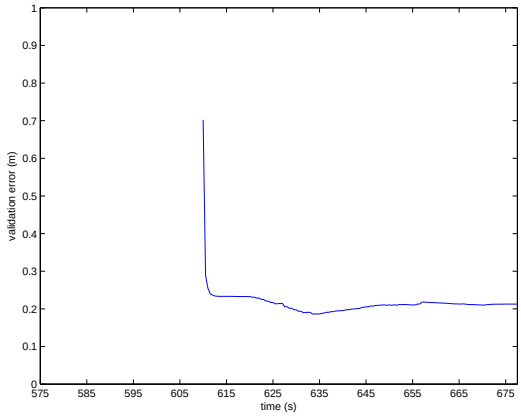
(b) Scenario 2



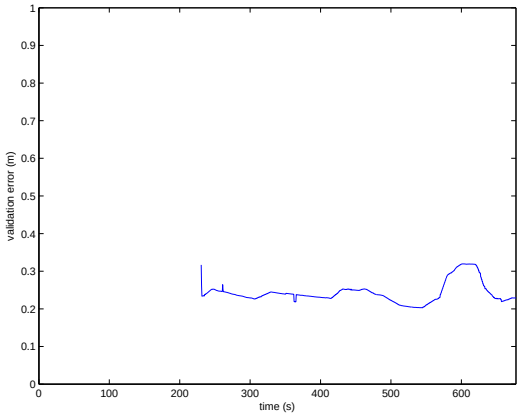
(c) Scenario 3



(d) Scenario 4



(e) Scenario 5



(f) Full timeline

Figure 6-8: Plots of validation accuracy versus time for ANN and DSET/ANN methods under 1 meter radar data.

6.4 Conclusion

This chapter built upon the technical successes of the previous one by proposing a novel algorithm for predicting UAV path data through a combination of the four methods described in Chapter 5. This algorithm improved prediction accuracy by dynamically switching among the four candidate methods over time based on which one is predicted to produce the least amount of validation error. When evaluating individual UAVs as well as the averages thereof, the proposed method outperforms the individual methods in most cases. For multi-UAV input and output, the lack of crossover points in the previous chapter result in negligible difference in accuracy between EKF/UKF and the proposed method. Overall, however, this algorithm demonstrates a further step forward in enhancing accurate prediction of autonomous UAV data, thereby incorporating another milestone in the enhancement of unmanned flight safety.

Chapter 7

Conclusions and Future Work

The goal of this dissertation was to bridge multiple emerging research fields into a seamless two-prong approach to flight safety aided by the many unique and powerful tools offered by machine learning. More specifically, this research brought a fresh new perspective to flight safety from viewing it from the perspectives of both manned and unmanned flight. The basis, background, and motivations of this were introduced in Chapter 1. From there, a broad comprehensive literature review was given followed by a thorough four-step technical approach. From these fundamental steps, the following contributions have been noted.

Chapter 2 consisted of the full literature review, spanning a wide array of fundamental building blocks for the technical contributions of this dissertation. This included previous approaches to examining flight safety as the roles of machine learning in assessing cognitive workload (including for manned flight) and in predicting UAV data (hence unmanned flight). The concepts of autonomy, cognitive workload, and data fusion were discussed.

Chapter 3 examined the portion of the research examining manned side of flight safety. This was performed using the concept of cognitive workload. More specifically, this chapter conducted an analysis of alternatives of the different machine learning techniques used in assessing cognitive workload. To do so, each technique

was tested under a variety of experimental parameters on four relevant datasets. From there, the advantages and disadvantages of each method under each dataset were discussed. Overall, these results provide insight into the ability to apply the most effective machine learning method under any given scenario to optimize the assessment of cognitive workload.

Chapter 4 extended upon the work in the previous chapter by presenting the development of two adaptive algorithms for dynamically allocating the most optimal machine learning technique given the attributes of a selected dataset. One statically selected a method, while the other dynamically selected methods at different crossover points across data samples. This method was tested on the four datasets provided in the previous chapter. Overall, the results indicated that the pilot-based dataset was the most ideal for the dynamic selection process, while others were better enhanced by the single selection algorithm. By being able to maximize metrics such as accuracy and efficiency, this algorithm has the potential for use in early detection of cognitive overload or underload of a pilot, thereby minimizing human factor based difficulties and maximizing this category of flight safety.

From there, approaches to the unmanned side of flight safety commenced in Chapter 5. More specifically, this chapter described a process for enhancing the role of machine learning in predicting UAV data by incorporating data fusion. A novel algorithm was thereby proposed that incorporated Dempster-Shafer Evidence Theory in combination with an artificial neural network to improve near term prediction of a UAV's navigational coordinates. When comparing the technique against a standard neural network as well as an extended Kalman filter and unscented Kalman filter, the results show a significant improvement in overall prediction accuracy.

Finally, Chapter 6 extended the achievements of the previous chapter by presenting an adaptive system for dynamically selecting prediction methods for optimal prediction accuracy in real time. This method was tested on the dataset from the

previous chapter under the same variety of experimental parameters. Within seconds of each UAV timeline, prediction error never exceeded 0.4 meters. This achievement thereby brings promising potential for safer unmanned flight safety in a real-time system.

7.1 Future Work

Due to the wide breadth of areas covered in this research, there are many possible paths for future research endeavors.

In Chapter 3, the clearest future direction would be the testing of more machine learning techniques such as Gaussian processes or gradient tree boosting. This could also include the addition of unsupervised methods such as restricted Boltzmann machines or unsupervised nearest neighbors. Another path could include the acquisition of more datasets for deeper comparisons, including data that approaches workload as a more numerical quantity rather than a label-based one. This can lead to the comparison of regression techniques in place of classifications.

In Chapter 4, the characteristics for ML method selection were sufficient for a small arsenal of four datasets, but if more datasets were to be added, there would need to be more stringent criteria for initial selection of a technique. In addition, a computational limitation with the second algorithm was the need to evaluate every possible method at each time step, thereby resulting in a significant amount of computational complexity. Thus, a logical future step would be to modify the algorithm to accurately predict when a crossover point between algorithms may occur. Thus, predictions could be made for one algorithm per time step when no nearby crossover is anticipated, thereby saving computational cost.

As for Chapter 5, possible future steps can include new ways of applying DSET to inputs and outputs as well as trying alternate data fusion techniques (e.g. Bayesian

Inference) or hybrid methods (e.g. fuzzy DSET, fuzzy rough set theory). Other potential directions include adding more types of sensory data, such as detection of terrain or obstacles. These algorithms could also be expanded to cover different types of predictions, such as UAV decisions themselves rather than the navigational coordinates that follow such decisions. This can be accomplished by focusing predictions to take the form of classifications rather than regressions.

With Chapter 6, the computational limitation exists with the proposed algorithm as with the dynamic selection algorithm in Chapter 4. That is, the need to evaluate every method at each time step can be highly complex computationally. Thus, the future step described in the Chapter 4 future work could apply here just as well. This is especially vital when transitioning from simplified theory-based simulations to practical real-time applications, particularly when interfacing with UAV hardware.

While the above are chapter-specific, others can be applied to the more general scope of this dissertation. For instance, because this research focused primarily on the two ends of the manned-unmanned flight spectrum, additional datasets can be included in the future to provide additional data points. One that could be particularly useful is of the physiological data and cognitive workload of remote UAV pilots. This would provide a unique bridge between the two categories of flight safety and could be applied to any of the previous four chapters when paired with UAV sensory data. In the public domain, such a dataset does not appear to be easily available. Thus, such a pursuit would involve either creating new data or contacting research groups that could potentially have such resources.

References

[1] R. Parasuraman, T. B. Sheridan, and C. D. Wickens, "A model for types and levels of human interaction with automation," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 30, no. 3, pp. 286–297, May 2000.

[2] A. M. Gibbons, T. L. von Thaden, and D. A. Wiegmann, "Development and initial validation of a survey for assessing safety culture within commercial flight operations," *The International Journal of Aviation Psychology*, vol. 16, no. 2, pp. 215–238, 2006.

[3] G. Boschert, M. Dery, and J. Hustwit, "Pilot safety survey 2005: What pilots are saying," *Air Medical Journal*, vol. 26, no. 1, pp. 34–37, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1067991X06002215

[4] H. Remawi, P. Bates, and I. Dix, "The relationship between the implementation of a safety management system and the attitudes of employees towards unsafe acts in aviation," *Safety Science*, vol. 49, no. 5, pp. 625–632, 2011.

[5] R. D. Brown and S. M. Galster, "Effects of reliable and unreliable automation on subjective measures of mental workload, situation awareness, trust and confidence in a dynamic flight task," *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 48, no. 1, pp. 147–151, 2004.

[6] Y.-H. Chang and C.-H. Yeh, "A survey analysis of service quality for domestic

airlines," *European Journal of Operational Research*, vol. 139, no. 1, pp. 166–177, 2002.

[7] P. J. Sherman, R. L. Helmreich, and A. C. Merritt, "National culture and flight deck automation: results of a multination survey," *International Journal of Aviation Psychology*, vol. 7, no. 4, pp. 311–329, 1997.

[8] M. Rudisill, "Line pilots' attitudes about and experience with flight deck automation results of an international survey and proposed guidelines," in *Proceedings of the 8th International Symposium on Aviation Psychology*, 1995.

[9] K. Funk, B. Lyall, J. Wilson, R. Vint, M. Niemczyk, C. Suroteguh, and G. Owen, "Flight deck automation issues," *The International Journal of Aviation Psychology*, vol. 9, no. 2, pp. 109–123, 1999.

[10] T. Halverson, J. Estepp, J. Christensen, and J. Monnin, "Classifying workload with eye movements in a complex task," *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 56, no. 1, pp. 168–172, 2012.

[11] W. Williams and M. Harris, "The challenges of flight-testing unmanned air vehicles," *Systems Engineering, Test and Evaluation Conference*, October 2002.

[12] B. M. Albaker and N. A. Rahim, "Unmanned aircraft collision detection and resolution: Concept and survey," in *2010 5th IEEE Conference on Industrial Electronics and Applications*, June 2010, pp. 248–253.

[13] N. A. of Sciences Engineering and Medicine, *Human-Automation Interaction Considerations for Unmanned Aerial System Integration into the National Airspace System: Proceedings of a Workshop*, N. Haller, Ed. Washington, DC: The National Academies Press, 2018.

[14] A. Hobba and H. R. Stanley, "Human factors in the maintenance of unmanned aircraft," *Unmanned Aerial Vehicles Human Factors, Program Review*, 2005.

[15] J. S. McCarley and C. D. Wickens, "Human factors concerns in uav flight," 2004.

[16] P. J. Hardin and R. R. Jensen, "Small-scale unmanned aerial vehicles in environmental remote sensing: Challenges and opportunities," *GIScience & Remote Sensing*, vol. 48, no. 1, pp. 99–111, 2011.

[17] G. Campbell and R. Lahey, "A survey of serious aircraft accidents involving fatigue fracture," *International Journal of Fatigue*, vol. 6, no. 1, pp. 25–30, 1984. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0142112384900057

[18] E. Wycoff and J. Holley, "Effects of flight attendants' touch upon airline passengers' perceptions of the attendant and the airline," vol. 71, pp. 932–934, 01 1990.

[19] D. O'Hare and D. Chalmers, "The incidence of incidents: A nationwide study of flight experience and exposure to accidents and.." *The International Journal of Aviation Psychology*, vol. 9, no. 1, pp. 1–18, 1999.

[20] J. B. Sexton, J. R. Klinect, and R. Helmreich, "The link between safety attitudes and observed performance in flight operations," in *Proceedings of the Eleventh International Symposium on Aviation Psychology*. Ohio State University Columbus, OH, 2001, pp. 7–13.

[21] Y.-H. Chang and C.-H. Yeh, "A new airline safety index," *Transportation Research Part B: Methodological*, vol. 38, no. 4, pp. 369–383, 2004. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S019126150300047X

[22] J. V. Belle, J. Shamoun-Baranes, E. V. Loon, and W. Bouten, "An operational model predicting autumn bird migration intensities for flight safety," *Journal of Applied Ecology*, vol. 44, no. 4, pp. 864–874, 2007.

[23] C. Macrae, "Making risks visible: Identifying and interpreting threats to airline flight safety," *Journal of Occupational and Organizational Psychology*, vol. 82, no. 2, pp. 273–293.

[24] P. O'Connor, A. O'Dea, Q. Kennedy, and S. E. Buttrey, "Measuring safety climate in aviation: A review and recommendations for the future," *Safety Science*, vol. 49, no. 2, pp. 128–138, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925753510002493

[25] B. Dhillon, "Human errors: A review," *Microelectronics Reliability*, vol. 29, no. 3, pp. 299–304, 1989. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0026271489906124

[26] B. Kirwan, "Human error identification in human reliability assessment. part 1: Overview of approaches," *Applied Ergonomics*, vol. 23, no. 5, pp. 299–318, 1992.

[27] ——, "Human error identification in human reliability assessment. part 2: Detailed comparison of techniques," *Applied Ergonomics*, vol. 23, no. 6, pp. 371–381, 1992.

[28] G. Conway, N. A Mode, M. D Berman, S. Martin, and A. Hill, "Flight safety in alaska: Comparing attitudes and practices of high- and low-risk air carriers," vol. 76, pp. 52–57, 02 2005.

[29] J. Caldwell and S. Gilreath, "A survey of aircrew fatigue in a sample of us army aviation personnel," vol. 73, pp. 472–480, 05 2002.

[30] M. R. Rosekind, P. H. Gander, L. J. Connell, and E. L. Co, "Crew factors in flight operations x, alertness management in flight operations," 1999. [Online]. Available: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19990116851.pdf

[31] E. L. Co, K. B. Gregory, J. M. Johnson, and M. R. Rosekind, "Crew factors in flight operations xi, a survey of fatigue factors in regional airline operations," 1999. [Online]. Available: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20000032967.pdf

[32] M. R. Rosekind, K. B. Gregory, E. L. Co, D. L. Miller, and D. F. Dinges, "Crew factors in flight operations xii, a survey of sleep quantity and quality in on-board crew rest facilities," 2000. [Online]. Available: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20010039028.pdf

[33] M. R. Rosekind, E. L. Co, K. B. Gregory, and D. L. Miller, "Crew factors in flight operations xiii, a survey of fatigue factors in corporate/executive aviation operations," 2000. [Online]. Available: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20010038243.pdf

[34] A. K. Emo, G. Matthews, and G. J. Funke, "The slow and the furious: Anger, stress and risky passing in simulated traffic congestion," *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 42, pp. 1–14, 2016.

[35] G. Matthews, A. K Emo, G. Funke, M. Zeidner, R. Roberts, P. Costa, and R. Schulze, "Emotional intelligence, personality, and task-induced stress," vol. 12, pp. 96–107, 07 2006.

[36] J. B. West, "A strategy for in-flight measurements of physiology of pilots of high-performance fighter aircraft," *Journal of Applied Physiology*, vol. 115, pp. 145–149, 2013.

[37] L. D. Loukopoulos, R. K. Dismukes, D. Loukopoulos, R. K. Dismukes, and I. Barshi, "Concurrent task demands in the cockpit: Challenges and vulnerabilities in routine flight operations."

[38] M. Funke, J. Warm, G. Matthews, M. Riley, V. Finomore, G. Funke, B. Knott, and M. Vidulich, "A comparison of cerebral hemovelocity and blood oxygen saturation levels during vigilance performance," *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 54, no. 1, pp. 1345–1349, 2010.

[39] J. A. Duley, S. M. Galster, and R. Parasuraman, "Information manager for determining data presentation preferences in future enroute air traffic management," *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 42, no. 1, pp. 47–51, 1998.

[40] H. A. Abbass, J. Tang, R. Amin, M. Ellejmi, and S. Kirby, "Augmented cognition using real-time eeg-based adaptive strategies for air traffic control," 2014.

[41] G. B. Chatterji and B. Sridhar, "Neural network based air traffic controller workload prediction," in *Proceedings of the 1999 American Control Conference*, 1999, pp. 2620–2624.

[42] J. B. Brookings, G. F. Wilson, and C. R. Swain, "Psychophysiological responses to changes in workload during simulated air traffic control," *Biological Psychology*, vol. 42, no. 3, pp. 361–377, 1996, psychophysiology of Workload.

[43] D. Gianazza, "Forecasting workload and airspace configuration with neural networks and tree search methods," *Artificial Intelligence*, vol. 174, no. 7, pp. 530–549, 2010.

[44] C. Kelleher and S. McGilloway, "Survey finds high levels of work-related stress among flight attendants," *Cabin Crew Safety*, vol. 40, no. 6, pp. 1–5, 2005.

[45] A. Sanders, "Towards a model of stress and human performance," *Acta Psychologica*, vol. 53, no. 1, pp. 61–97, 1983.

[46] G. R. J. Hockey, "Compensatory control in the regulation of human performance under stress and high workload: A cognitive-energetical framework," *Biological Psychology*, vol. 45, no. 1, pp. 73–93, 1997, mental Resources: Intensive and Selective Aspects.

[47] M. Mendl, "Performing under pressure: stress and cognitive function," *Applied Animal Behaviour Science*, vol. 65, no. 3, pp. 221–244, 1999.

[48] C. Berka, D. J. Levendowski, M. N. Lumicao, A. Yau, G. Davis, V. T. Zivkovic, R. E. Olmstead, P. D. Tremoulet, and P. L. Craven, "Eeg correlates of task engagement and mental workload in vigilance, learning, and memory tasks," *Aviation Space and Environmental Medicine*, vol. 78, no. 5, pp. B231–B234, 2007.

[49] G. Durantin, J.-F. Gagnon, S. Tremblay, and F. Debais, "Using near infrared spectroscopy and heart rate variability to detect mental overload," *Behavioural Brian Search*, vol. 259, pp. 16–23, 2014.

[50] M. Niaz and R. H. Logie, "Working memory, mental capacity and science education: towards an understanding of the 'working memory overload hypothesis'," *Oxford Review of Education*, vol. 19, no. 4, pp. 511–525, 1993.

[51] G. J. Funke and S. M. Galster, "The effects of cognitive processing load and collaboration technology on team performance in a simulated command and control environment," *International Journal of Industrial Ergonomics*, vol. 39, no. 3, pp. 541–547, 2009, selected papers from ECCE 2007, the 25th Anniversary Conference of the European Conference on Cognitive Ergonomics.

[52] J. Lenneman and R. Backs, "Cardiac autonomic control during simulated driving with a concurrent verbal working memory task," vol. 51, pp. 404–418, 06 2009.

[53] B. Reimer and B. Mehler, "The impact of cognitive workload on physiological arousal in young adult drivers: a field study and simulation validation," *Ergonomics*, vol. 54, no. 10, pp. 932–942, 2011.

[54] C. Collet, A. Clarion, M. Morel, A. Chapon, and C. Petit, "Physiological and behavioural changes associated to the management of secondary tasks while driving," *Applied Ergonomics*, vol. 40, no. 6, pp. 1041–1046, 2009, psychophysiology in Ergonomics.

[55] M. B. Dillard, J. S. Warm, G. J. Funke, M. E. Funke, J. Victor S. Finomore, G. Matthews, T. H. Shaw, and R. Parasuraman, "The sustained attention to response task (sart) does not promote mindlessness during vigilance performance," *Human Factors*, vol. 56, no. 8, pp. 1364–1379, 2014.

[56] A. J. Strang, C. Best, and G. J. Funke, "Heart rate correlates of mental workload in a large-scale air-combat simulation training exercise," *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 58, no. 1, pp. 2325–2329, 2014.

[57] A. Henelius, K. Hirvonen, A. Holm, J. Korpela, and K. Muller, "Mental workload classification using heart rate metrics," in *2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, Sept 2009, pp. 1836–1839.

[58] G. F. Wilson and C. A. Russell, "Real-time assessment of mental workload using psychophysiological measures and artificial neural networks," *Human Factors*, vol. 45, no. 4, pp. 635–644, 2016.

[59] C. Dussault, J.-C. Jouanin, M. Philippe, and C. Guezennec, "Eeg and ecg changes during simulator operation reflect mental workload and vigilance," vol. 76, pp. 344–351, 04 2005.

[60] S. W. Hincks, D. Afergan, and R. J. K. Jacob, "Using fnirs for real-time cognitive workload assessment," in *Foundations of Augmented Cognition: Neuroergonomics and Operational Neuroscience*, D. D. Schmorrow and C. M. Fidopiastis, Eds. Cham: Springer International Publishing, 2016, pp. 198–208.

[61] M. P. Çakır, M. Vural, S. Ö. Koç, and A. Toktaş, "Real-time monitoring of cognitive workload of airline pilots in a flight simulator with fnir optical brain imaging technology," in *Foundations of Augmented Cognition: Neuroergonomics and Operational Neuroscience*, D. D. Schmorrow and C. M. Fidopiastis, Eds. Cham: Springer International Publishing, 2016, pp. 147–158.

[62] A. Kramer, "Physiological metrics of mental workload: A review of recent progress," 07 1990.

[63] J. C. Christensen, J. R. Estepp, G. F. Wilson, and C. A. Russell, "The effects of day-to-day variability of physiological data on operator functional state classification," *NeuroImage*, vol. 59, no. 1, pp. 57–63, 2012, neuroergonomics: The human brain in action and at work.

[64] G. Funke, B. Knott, V. Mancuso, A. Strang, J. Estepp, L. Menke, R. Brown, A. Dukes, and B. Miller, "Evaluation of subjective and eeg-based measures of mental workload," in *Communications in Computer and Information Science*, vol. 373, 07 2013, pp. 412–416.

[65] P. Bashivan, I. Rish, M. Yeasin, and N. Codella, "Learning representations from eeg with deep recurrent-convolutional neural networks," in *4th International Conference on Learning Representations*, 2015.

[66] W. Zheng and B. Lu, "Investigating critical frequency bands and channels for eeg-based emotion recognition with deep neural networks," *IEEE Transactions on Autonomous Mental Development*, vol. 7, no. 3, pp. 162–175, 2015.

[67] A. Natarajan, K. S. Xu, and B. Eriksson, "Detecting divisions of the autonomic nervous system using wearables," in *IEEE 38th Annual International Conference of the Engineering in Medicine and Biology Society (EMBC)*, 2016, pp. 5761–5764.

[68] J. Veltman and A. W. K. Gaillard, "Physiological workload reactions to increasing levels of task difficulty," vol. 41, pp. 656–669, 06 1998.

[69] A. Roscoe, "Assessing pilot workload. why measure heart rate, hrv and respiration?" *Biological Psychology*, vol. 34, no. 2, pp. 259–287, 1992, special Issue Cardiorespiratory Measures and thier Role in Studies of Performance.

[70] Y.-H. Lee and B.-S. Liu, "Inflight workload assessment: Comparison of subjective and physiological measurements," vol. 74, pp. 1078–1084, 11 2003.

[71] C. Malsbeurg, "Frank rosenblatt: Principles of neurodynamics: Perceptrons and the theory of brain mechanisms," in *Proceedings of the First Trimester Meeting on Brain Theory*, pp. 245–248.

[72] M. Mavrovouniotis and S. Yang, "Evolving neural networks using ant colony optimization with pheromone trail limits," in *2013 13th UK Workshop on Computational Intelligence (UKCI)*, Sept 2013, pp. 16–23.

[73] G. F. Wilson, J. Estepp, and I. Davis, "A comparison of performance and psychophysiological classification of complex task performance," vol. 53, pp. 141–145, 10 2009.

[74] G. F. Wilson and C. A. Russell, "Real-time assessment of mental workload using psychophysiological measures and artificial neural networks," *Human Factors*, vol. 45, no. 4, pp. 635–644, 2003.

[75] C. L. Baldwin and B. N. Penaranda, "Adaptive training using an artificial neural network and eeg metrics for within- and cross-task workload classification," *NeuroImage*, vol. 59, no. 1, pp. 48–56, 2012, neuroergonomics: The human brain in action and at work.

[76] T. I. Laine, K. W. Bauer, J. W. Lanning, C. A. Russell, and G. F. Wilson, "Selection of input features across subjects for classifying crewmember workload using artificial neural networks," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 32, no. 6, pp. 691–704, Nov 2002.

[77] J. Hennrich, C. Herff, D. Heger, and T. Schultz, "Investigating deep learning for fnirs based bci," in *37th Annual International Conference of the Engineering in Medicine and Biology Society (EMBC)*, 2015, pp. 2844–2847.

[78] S. Sarkar, K. Reddy, A. Dorgan, C. Fidopiastis, and M. Giering, "Wearable eeg-based activity recognition in phm-related service environment via deep learning," *International Journal of Prognostics and Health Management*, 2016.

[79] R. G. Hefron, B. J. Borghetti, J. C. Christensen, and C. M. S. Kabban, "Deep long short-term memory structures model temporal dependencies improving cognitive workload estimation," *Pattern Recogn. Lett.*, vol. 94, no. C, pp. 96–104, Jul. 2017.

[80] T. Juhaniak, P. Hlavac, R. Móro, J. Simko, and M. Bieliková, "Pupillary response: Removing screen luminosity effects for clearer implicit feedback," in *UMAP*, 2016.

[81] C. Tran, A. Abraham, and L. Jain, "Decision support systems using hybrid neurocomputing," *Neurocomputing*, vol. 61, pp. 85–97, 2004, hybrid Neurocomputing: Selected Papers from the 2nd International Conference on Hybrid Intelligent Systems.

[82] M. D. Ziegler, A. Kraft, M. Krein, L.-C. Lo, B. Hatfield, W. Casebeer, and B. Russell, "Sensing and assessing cognitive workload across multiple tasks," in *Foundations of Augmented Cognition: Neuroergonomics and Operational Neuroscience*, 2016, pp. 440–450.

[83] M. Hannula, K. Huttunen, J. Koskelo, T. Laitinen, and T. Leino, "Comparison between artificial neural network and multilinear regression models in an evaluation of cognitive workload in a flight simulator," *Computers in Biology and Medicine*, vol. 38, no. 11, pp. 1163–1170, 2008.

[84] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, pp. 273–297, 1995.

[85] L. Jin, Q. Niu, H. Hou, H. Xian, Y. Wang, and D. Shi, "Driver cognitive distraction detection using driving performance measures," *Discrete Dynamics in Nature and Society*, 2012.

[86] J. Son, H. Oh, and M. Park, "Identification of driver cognitive workload using support vector machines with driving performance, physiology and eye movement in a driving simulator," *International Journal of Precision Engineering and Manufacturing*, vol. 14, no. 8, pp. 1321–1327, 2013.

[87] F. Putze, J. Jarvis, and T. Schultz, "Multimodal recognition of cognitive workload for multitasking in the car," in *2010 20th International Conference on Pattern Recognition (ICPR)*, 2010, pp. 3748–3751.

[88] Y. Liang, M. Reyes, and J. Lee, "Real-time detection of driver cognitive distraction using support vector machines," *IEEE Transactions on Intelligent Transportation Systems*, vol. 8, no. 2, pp. 340–350, 2007.

[89] Z. Yin and J. Zhang, "Identification of temporal variations in mental workload using locally-linear-embedding-based eeg feature reduction and support-vector-machine-based clustering and classification techniques," *Computer Methods and Programs in Biomedicine*, vol. 115, no. 3, pp. 119–134, 2014.

[90] E. T. Solovey, M. Zec, E. A. Garcia Perez, B. Reimer, and B. Mehler, "Classifying driver workload using physiological and driving performance data: Two field studies," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '14. New York, NY, USA: ACM, 2014, pp. 4057–4066.

[91] O. Kramer, *K-Nearest Neighbors*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 13–23.

[92] T. K. Calibo, J. A. Blanco, and S. L. Firebaugh, "Cognitive stress recognition," in *2013 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, May 2013, pp. 1471–1475.

[93] A. Girouard, "Distinguishing difficulty levels with non-invasive brain activity measurements," in *Human-Computer Interaction – INTERACT 2009*, 2009, pp. 440–452.

[94] L. Breiman, J. Friedman, C. Stone, and R. Olshen, *Classification and Regression Trees*, ser. The Wadsworth and Brooks-Cole statistics-probability series. Taylor & Francis, 1984.

[95] A. Liaw, M. Wiener *et al.*, "Classification and regression by randomforest," *R news*, vol. 2, no. 3, pp. 18–22, 2002.

[96] M. R. ENDSLEY and D. K. Kaber, "Level of automation effects on performance, situation awareness and workload in a dynamic control task," *Ergonomics*, vol. 42, no. 3, pp. 462–492, 1999.

[97] D. Bruemmer, D. D. Dudenhoeffer, and J. Marble, "Dynamic-autonomy for urban search and rescue." in *AAAI Mobile Robot Competition 2002, Papers from the AAAI Workshop*, July 2002, pp. 33–37.

[98] M. DESAI, "Sliding scale autonomy and trust in human-robot interaction," 03 2018.

[99] T. Helldin, G. Falkman, M. Riveiro, and S. Davidsson, "Presenting system uncertainty in automotive uis for supporting trust calibration in autonomous driving," in *Proceedings of the 5th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, ser. AutomotiveUI '13. New York, NY, USA: ACM, 2013, pp. 210–217.

[100] J. C. Christensen and J. R. Estepp, "Coadaptive aiding and automation enhance operator performance," *Human Factors*, vol. 55, no. 5, pp. 965–975, 2013.

[101] J. M. Beer, A. D. Fisk, and W. A. Rogers, "Toward a framework for levels of robot autonomy in human-robot interaction," *J. Hum.-Robot Interact.*, vol. 3, no. 2, pp. 74–99, Jul. 2014.

[102] B. Parke, E. Chevalley, P. Lee, F. Omar, J. M. Kraut, K. Gonter, A. Borade, C. Gabriel, N. Bienert, C. Lin, H. S. Yoo, D. Rein-Weston, and E. Palmer, "Coordination between sectors in shared airspace operations," in *2014 IEEE/AIAA 33rd Digital Avionics Systems Conference (DASC)*, 2014, pp. 7C3–1–7C3–12.

[103] E. Chevalley, B. Parke, P. Lee, F. Omar, H. Lee, N. Bienert, J. Kraut, and E. Palmer, "Scheduling and separating departures crossing arrival flows in

shared airspace," in *2013 IEEE/AIAA 32nd Digital Avionics Systems Conference (DASC)*, Oct 2013, pp. 1D4–1–1D4–16.

[104] D. Rein-Weston and E. Chevalley, "Development of a route crossing tool for shared airspace environments," in *2014 IEEE/AIAA 33rd Digital Avionics Systems Conference (DASC)*, Oct 2014, pp. 1–24.

[105] C. Kurkcu, H. Erhan, and S. Umut, "Human factors concerning unmanned aircraft systems in future operations," *Journal of Intelligent & Robotic Systems*, vol. 65, no. 1, pp. 63–72, Jan 2012.

[106] J. M. Peschel and R. R. Murphy, "On the human 2013; machine interaction of unmanned aerial system mission specialists," *IEEE Transactions on Human-Machine Systems*, vol. 43, no. 1, pp. 53–62, Jan 2013.

[107] S. Guznov, G. Matthews, G. Funke, and A. Dukes, "Use of the roboflag synthetic task environment to investigate workload and stress responses in uav operation," *Behavior Research Methods*, vol. 43, no. 3, pp. 771–780, Sep 2011.

[108] N. Cooke, "Human factors of remotely operated vehicles," in *Proceedings of the Human Factors and Ergonomics Society*, 2006, pp. 166–169.

[109] P. Jasper, C. Sibley, and J. Coyne, "Using heart rate variability to assess operator mental workload in a command and control simulation of multiple unmanned aerial vehicles," *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 60, no. 1, pp. 1125–1129, 2016.

[110] S.-L. Chen and Y. Jen, "Data fusion neural network for tool condition monitoring in CNC milling machining," *International Journal of Machine Tools and Manufacture*, vol. 40, no. 3, pp. 381–400, 2000.

[111] R. W. Wohleber, G. Matthews, G. J. Funke, and J. Lin, "Considerations in physiological metric selection for online detection of operator state: A case study," in *Foundations of Augmented Cognition: Neuroergonomics and Operational Neuroscience*, D. D. Schmorrow and C. M. Fidopiastis, Eds. Cham: Springer International Publishing, 2016, pp. 428–439.

[112] L. Lin, Q. Sun, S. Wang, and F. Yang, "A geographic mobility prediction routing protocol for ad hoc UAV network," in *2012 IEEE Globecom Workshops*, Dec 2012, pp. 1597–1602.

[113] B. T. Clough, "Unmanned aerial vehicles: Autonomous control challenges, a researcher's perspective," *Journal of Aerospace Computing, Information, and Communication*, vol. 2, no. 8, pp. 327–347, 2005.

[114] S. Karim and C. Heinze, "Experiences with the design and implementation of an agent-based autonomous uav controller," in *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, ser. AAMAS '05. New York, NY, USA: ACM, 2005, pp. 19–26.

[115] T. Tomic, K. Schmid, P. Lutz, A. Domel, M. Kassecker, E. Mair, I. L. Grixa, F. Ruess, M. Suppa, and D. Burschka, "Toward a fully autonomous uav: Research platform for indoor and outdoor urban search and rescue," *IEEE Robotics Automation Magazine*, vol. 19, no. 3, pp. 46–56, Sept 2012.

[116] A. Bürkle, F. Segor, and M. Kollmann, "Towards autonomous micro uav swarms," *Journal of Intelligent & Robotic Systems*, vol. 61, no. 1, pp. 339–353, Jan 2011. [Online]. Available: https://doi.org/10.1007/s10846-010-9492-x

[117] K. P. Valavanis and G. J. Vachtsevanos, *Future of Unmanned Aviation*. Dordrecht: Springer Netherlands, 2015, pp. 2993–3009.

[118] S. Bosch, S. Lacroix, and F. Caballero, "Autonomous detection of safe landing areas for an uav from monocular images," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2006, pp. 5522–5527.

[119] P. Fabiani, V. Fuertes, A. Piquereau, R. Mampey, and F. Teichteil-Königsbuch, "Autonomous flight and navigation of vtol uavs: from autonomy demonstrations to out-of-sight flights," *Aerospace Science and Technology*, vol. 11, no. 2, pp. 183–193, 2007.

[120] F. Nex and F. Remondino, "Uav for 3d mapping applications: a review," *Applied Geomatics*, vol. 6, no. 1, pp. 1–15, Mar 2014.

[121] S. Franklin, T. Madl, S. D'Mello, and J. Snaider, "Lida: A systems-level architecture for cognition, emotion, and learning," *IEEE Transactions on Autonomous Mental Development*, vol. 6, no. 1, pp. 19–41, March 2014.

[122] J. E. Laird, A. Newell, and P. S. Rosenbloom, "Soar: An architecture for general intelligence," *Artificial Intelligence*, vol. 33, no. 1, pp. 1–64, 1987.

[123] A. M. Nuxoll and J. E. Laird, "Enhancing intelligent agents with episodic memory," *Cognitive Systems Research*, vol. 17-18, no. Supplement C, pp. 34–48, 2012.

[124] J. R. Anderson, "Act: A simple theory of complex cognition," *American Psychologist*, vol. 51, no. 4, pp. 355–365, 1996.

[125] E. Salas and S. Fiore, "Team cognition: Understanding the factors that drive process and performance," 01 2004.

[126] B. F. Malle, S. Guglielmo, and A. E. Monroe, "A theory of blame," *Psychological Inquiry*, vol. 25, no. 2, pp. 147–186, 2014.

[127] D. G. Tecuci, "A generic memory module for events," Ph.D. dissertation, Austin, TX, USA, 2007, aAI3277673.

[128] W. Wang, B. Subagdja, A. H. Tan, and Y.-S. Tan, "A self-organizing multi-memory system for autonomous agents," in *The 2012 International Joint Conference on Neural Networks (IJCNN)*, June 2012, pp. 1–8.

[129] D. Stachowicz and G. J. M. Kruijff, "Episodic-like memory for cognitive robots," *IEEE Transactions on Autonomous Mental Development*, vol. 4, no. 1, pp. 1–16, March 2012.

[130] B. Subagdja, W. Wang, A.-H. Tan, Y.-S. Tan, and L.-N. Teow, "Memory formation, consolidation, and forgetting in learning agents," in *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems - Volume 2*, ser. AAMAS '12. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2012, pp. 1007–1014. [Online]. Available: http://dl.acm.org/citation.cfm?id=2343776.2343841

[131] S. Jockel, M. Weser, D. Westhoff, and J. Zhang, "Towards an episodic memory for cognitive robots," 03 2018.

[132] D. Wang, A.-H. Tan, and C. Miao, "Modeling autobiographical memory in human-like autonomous agents," in *Proceedings of the 2016 International Conference on Autonomous Agents; Multiagent Systems*, ser. AAMAS '16. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2016, pp. 845–853.

[133] W. Wang, B. Subagdja, A. H. Tan, and J. A. Starzyk, "A self-organizing approach to episodic memory modeling," in *The 2010 International Joint Conference on Neural Networks (IJCNN)*, July 2010, pp. 1–8.

[134] R. Wood, P. Baxter, and T. Belpaeme, "A review of long-term memory in natural and synthetic systems," vol. 20, 01 2012.

[135] T. D. Kelley, "Robotic dreams: A computational justification for the post-hoc processing of episodic memories," *International Journal of Machine Consciousness*, vol. 06, no. 02, pp. 109–123, 2014.

[136] R. Kadlec and C. Brom, *DyBaNeM: Bayesian Episodic Memory Framework for Intelligent Virtual Agents*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 15–28.

[137] L. Macedo and A. Cardoso, "Assessing creativity: the importance of unexpected novelty," *Structure*, vol. 1, no. C2, p. C3, 2002.

[138] ——, "A model for generating expectations: the bridge between memory and surprise," in *IJCAI 2003 Workshop 3rd. Workshop on Creative Systems*, 2003, p. 2.

[139] ——, "The influence of the size of the episodic memory on the surprise-value of the creative agent's products," in *Proceedings of the First Workshop on Creative Systems-International Conference on Case-Based Reasoning*, 2001.

[140] S. L. Chu, E. A. Fedorovskaya, F. K. Quek, and J. Snyder, "The effect of familiarity on perceived interestingness of images," in *Human Vision and Electronic Imaging*, 2013, p. 86511C.

[141] N. Derbinsky and J. E. Laird, *Efficiently Implementing Episodic Memory*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 403–417.

[142] M. Kazemifard, N. Ghasem-Aghaee, B. L. Koenig, and T. I. Ören, "An emotion understanding framework for intelligent agents based on episodic and semantic

memories," *Autonomous Agents and Multi-Agent Systems*, vol. 28, no. 1, pp. 126–153, Jan 2014.

[143] W. Wang, B. Subagdja, A. Tan, and J. A. Starzyk, "Neural modeling of episodic memory: Encoding, retrieval, and forgetting," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 10, pp. 1574–1586, Oct 2012.

[144] E. Barakova and T. Lourens, "Spatial navigation based on novelty mediated autobiographical memory," in *Mechanisms, Symbols, and Models Underlying Cognition*, J. Mira and J. R. Álvarez, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 356–365.

[145] A. Y. Javaid, W. Sun, and M. Alam, "UAVSim: A simulation testbed for unmanned aerial vehicle network cyber security analysis," in *2013 IEEE Globecom Workshops (GC Wkshps)*, Dec 2013, pp. 1432–1436.

[146] S. R. Dixon, C. D. Wickens, and D. Chang, "Comparing quantitative model predictions to experimental data in multiple-UAV flight control," *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 47, no. 1, pp. 104–108, 2003.

[147] M. L. Cummings and P. J. Mitchell, "Predicting controller capacity in supervisory control of multiple UAVs," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 38, no. 2, pp. 451–460, March 2008.

[148] T. Dierks and S. Jagannathan, "Neural network output feedback control of a quadrotor UAV," in *2008 47th IEEE Conference on Decision and Control*, Dec 2008, pp. 3633–3639.

[149] J. Shin, H. J. Kim, and Y. Kim, "Adaptive support vector regression for UAV flight control," *Neural Networks*, vol. 24, no. 1, pp. 109–120, 2011.

[150] W. Hoburg and R. Tedrake, "System identification of post stall aerodynamics for UAV perching," 04 2009.

[151] L. Wallace, A. Lucieer, and C. S. Watson, "Evaluating tree detection and segmentation routines on very high resolution UAV LiDAR data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 12, pp. 7619–7628, Dec 2014.

[152] C. Gevaert, C. Persello, R. Sliuzas, and G. Vosselman, "Integration of 2D and 3D features from UAV imagery for informal settlement classification using multiple kernel learning," in *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, July 2016, pp. 1508–1511.

[153] A. Puri, K. P. Valavanis, and M. Kontitsis, "Statistical profile generation for traffic monitoring using real-time UAV based video data," in *2007 Mediterranean Conference on Control Automation*, June 2007, pp. 1–6.

[154] L. Baoan, L. Zhihua, and L. Xinjun, "Research of UAV engine fault prediction based on particle filter," in *2009 9th International Conference on Electronic Measurement Instruments*, Aug 2009, pp. 813–817.

[155] H. Jaenisch and J. Handley, "Automatic differential equation data modeling for UAV situational awareness," in *Society for Computer Simulation, Huntsville Simulation Conference*, 2003.

[156] A. Cherian, J. Andersh, V. Morellas, N. Papanikolopoulos, and B. Mettler, "Autonomous altitude estimation of a UAV using a single onboard camera," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2009, pp. 3900–3905.

[157] M. Rhudy, T. Larrabee, H. Chao, Y. Gu, and M. Napolitano, "UAV attitude, heading, and wind estimation using GPS/INS and an air data system," in *AIAA*

Guidance, Navigation, and Control (GNC) Conference, Guidance, Navigation, and Control and Co-located Conferences, August 2013.

[158] J. Woo, K. Son, T. Li, G. S. Kim, and I.-S. Kweon, "Vision-based UAV navigation in mountain area." in *MVA*, 2007, pp. 236–239.

[159] K. Dogancay, "UAV path planning for passive emitter localization," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, no. 2, pp. 1150–1166, APRIL 2012.

[160] D. Shen, G. Chen, J. B. Cruz, and E. Blasch, "A game theoretic data fusion aided path planning approach for cooperative UAV ISR," in *2008 IEEE Aerospace Conference*, March 2008, pp. 1–9.

[161] L.-C. Wang and C. Y. Wen, "Adaptive trajectory tracking for UAV guidance with Bayesian filtering," in *2010 5th IEEE Conference on Industrial Electronics and Applications*, June 2010, pp. 2311–2316.

[162] T. Zsedrovits, A. Zarándy, B. Vanek, T. Péni, J. Bokor, and T. Roska, "Collision avoidance for UAV using visual detection," in *2011 IEEE International Symposium of Circuits and Systems (ISCAS)*, May 2011, pp. 2173–2176.

[163] X. Wang, V. Yadav, and S. N. Balakrishnan, "Cooperative UAV formation flying with obstacle/collision avoidance," *IEEE Transactions on Control Systems Technology*, vol. 15, no. 4, July 2007.

[164] M. Rhudy, Y. Gu, J. Gross, and M. R. Napolitano, "Evaluation of matrix square root operations for UKF within a UAV GPS/INS sensor fusion application," vol. 2011, 12 2011.

[165] S. Adusumilli, D. Bhatt, H. Wang, V. Devabhaktuni, and P. Bhattacharya, "A novel hybrid approach utilizing principal component regression and random

forest regression to bridge the period of GPS outages," *Neurocomputing*, vol. 166, no. Supplement C, pp. 185–192, 2015.

[166] D. Bhatt, P. Aggarwal, V. Devabhaktuni, and P. Bhattacharya, "A novel hybrid fusion algorithm to bridge the period of GPS outages using low-cost INS," *Expert Systems with Applications*, vol. 41, no. 5, pp. 2166–2173, 2014.

[167] S. Adusumilli, D. Bhatt, H. Wang, P. Bhattacharya, and V. Devabhaktuni, "A low-cost INS/GPS integration methodology based on random forest regression," *Expert Systems with Applications*, vol. 40, no. 11, pp. 4653–4659, 2013.

[168] P. Aggarwal, D. Bhatt, V. Devabhaktuni, and P. Bhattacharya, "Dempster Shafer neural network algorithm for land vehicle navigation application," *Information Sciences*, vol. 253, no. Supplement C, pp. 26–33, 2013.

[169] D. Bhatt, P. Aggarwal, V. Devabhaktuni, and P. Bhattacharya, "A new source difference artificial neural network for enhanced positioning accuracy," *Measurement Science and Technology*, vol. 23, no. 10, p. 105101, 2012.

[170] A. Nemra and N. Aouf, "Robust INS/GPS sensor fusion for UAV localization using SDRE nonlinear filtering," *IEEE Sensors Journal*, vol. 10, no. 4, pp. 789–798, April 2010.

[171] C. S. Y. C.-S. Yoo and I. K. A. I.-K. Ahn, "Low cost GPS/INS sensor fusion system for UAV navigation," in *Digital Avionics Systems Conference, 2003. DASC '03. The 22nd*, vol. 2, Oct 2003, p. 8.A.1.

[172] A. Verma, C.-N. Wu, and V. Castelli, "Autonomous command and control system for UAV formation," 08 2003.

[173] G. Chen, D. Shen, C. Kwan, J. B. Cruz, and M. Kruger, "Game theoretic ap-

proach to threat prediction and situation awareness," in *2006 9th International Conference on Information Fusion*, July 2006, pp. 1–8.

[174] B. Yu, J. Giampapa, S. Owens, and K. Sycara, "An evidential model of multisensor decision fusion for force aggregation and classification," in *2005 7th International Conference on Information Fusion*, vol. 2, July 2005.

[175] O. Basir and X. Yuan, "Engine fault diagnosis based on multi-sensor information fusion using Dempster-Shafer evidence theory," *Information Fusion*, vol. 8, no. 4, pp. 379–386, 2007.

[176] M. Safizadeh and S. Latifi, "Using multi-sensor data fusion for vibration fault diagnosis of rolling element bearings by accelerometer and load cell," *Information Fusion*, vol. 18, no. Supplement C, pp. 1–8, 2014.

[177] H. Durrant-Whyte, M. Stevens, and E. Nettleton, "Data fusion in decentralised sensing networks," in *4th International Conference on Information Fusion*, 2001, pp. 302–307.

[178] N. Ghosh, Y. Ravi, A. Patra, S. Mukhopadhyay, S. Paul, A. Mohanty, and A. Chattopadhyay, "Estimation of tool wear during CNC milling using neural network-based sensor fusion," *Mechanical Systems and Signal Processing*, vol. 21, no. 1, pp. 466–479, 2007.

[179] D. A. Dornfeld and M. DeVries, "Neural network sensor fusion for tool condition monitoring," *CIRP Annals*, vol. 39, no. 1, pp. 101–105, 1990.

[180] J. Yuqin, W. Peixia, and L. Yue, "Study of manufacturing system based on neural network multi-sensor data fusion and its application," in *IEEE International Conference on Robotics, Intelligent Systems and Signal Processing, 2003. Proceedings. 2003*, vol. 2, Oct 2003, pp. 1022–1026 vol.2.

[181] Y. Zhang, J. Pulliainen, S. Koponen, and M. Hallikainen, "Application of an empirical neural network to surface water quality estimation in the Gulf of Finland using combined optical data and microwave data," *Remote Sensing of Environment*, vol. 81, no. 2, pp. 327–336, 2002.

[182] D. G. R. Loyola, "Automatic cloud analysis from polar-orbiting satellites using neural network and data fusion techniques," in *IGARSS 2004. 2004 IEEE International Geoscience and Remote Sensing Symposium*, vol. 4, Sept 2004, pp. 2530–2533 vol.4.

[183] P. Nelson and P. Palacharla, *Neural Network Model for Data Fusion in ADVANCE*. ASCE, 1993, pp. 237–293.

[184] X. Dai and S. Khorram, "Data fusion using artificial neural networks: a case study on multitemporal change analysis," *Computers, Environment and Urban Systems*, vol. 23, no. 1, pp. 19–31, 1999.

[185] C. Li, P. Heinemann, and R. Sherry, "Neural network and Bayesian network fusion models to fuse electronic nose and surface acoustic wave sensor data for apple defect detection," *Sensors and Actuators B: Chemical*, vol. 125, no. 1, pp. 301–310, 2007.

[186] T. Denoeux, "A neural network classifier based on Dempster-Shafer theory," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 30, no. 2, pp. 131–150, Mar 2000.

[187] L. Chin, "Application of neural networks in target tracking data fusion," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 30, no. 1, pp. 281–287, Jan 1994.

[188] S. Merigeault, M. Batariere, and J. N. Patillon, "Data fusion based on neural network for the mobile subscriber location," in *Vehicular Technology Conference*

Fall 2000. IEEE VTS Fall VTC2000. 52nd Vehicular Technology Conference (Cat. No.00CH37152), vol. 2, 2000, pp. 536–541 vol.2.

[189] H. T. Nguyen and E. A. Walker, "On decision making using belief functions," *Advances in the Dempster-Shafer theory of evidence*, pp. 311–330, 1994.

[190] E. Chiodo and G. Mazzanti, "Bayesian reliability estimation based on a Weibull stress-strength model for aged power system components subjected to voltage surges," *IEEE Transactions on Dielectrics and Electrical Insulation*, vol. 13, no. 1, pp. 146–159, Feb 2006.

[191] P. Limbourg and R. Savic, "Fault tree analysis in an early design stage using the Dempster-Shafer theory of evidence," 01 2007.

[192] G. Lipeng, J. Juan, and D. Laibin, "An improved fusion algorithm of evidence theory," in *Proceedings of 2011 Cross Strait Quad-Regional Radio Science and Wireless Technology Conference*, vol. 2, July 2011, pp. 1495–1498.

[193] I. Rish, "An empirical study of the naive bayes classifier," Tech. Rep., 2001.

[194] E. Chiodo, D. Lauria, and C. Pisani, "Bayes estimation of wind speed extreme values," in *3rd Renewable Power Generation Conference (RPG 2014)*, Sept 2014, pp. 1–7.

[195] E. Chiodo and D. Lauria, "Double stochastic analysis of batteries lifetime for electric vehicles operation," 2014.

[196] G. Shafer, "Perspectives on the theory and practice of belief functions," *International Journal of Approximate Reasoning*, vol. 4, no. 5, pp. 323–362, 1990.

[197] A. Ekaterina, W. Luther, G. Rebner, and P. Limbourg, "A verified MATLAB toolbox for the Dempster-Shafer theory," 01 2010.

[198] "Imprecise probability propagation toolbox (IPP toolbox)," 2012, Essen, Germany: University of Duisberg-Essen, Department of Information Logistics (https://www.uni-due.de/informationslogistik/ipptoolbox.php).

[199] F. Tonon, "Using random set theory to propagate epistemic uncertainty through a mechanical system," *Reliability Engineering & System Safety*, vol. 85, no. 1, pp. 169–181, 2004, alternative Representations of Epistemic Uncertainty.

[200] A. Al-Ani and M. A. Deriche, "A new technique for combining multiple classifiers using the dempster-shafer theory of evidence," *J. Artif. Intell. Res.*, vol. 17, pp. 333–361, 2002.

[201] C. Elkin, R. Kumarasiri, D. B. Rawat, and V. Devabhaktuni, "Localization in wireless sensor networks: A Dempster-Shafer evidence theoretical approach," *Ad Hoc Networks*, vol. 54, no. Supplement C, pp. 30–41, 2017.

[202] A. Clerentin, L. Delahoche, C. Pegard, and E. B. Gracsy, "A localization method based on two omnidirectional perception systems cooperation," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 2, 2000, pp. 1219–1224.

[203] P. Kasebzadeh, G. S. Granados, and E. S. Lohan, "Indoor localization via WLAN path-loss models and Dempster-Shafer combining," in *International Conference on Localization and GNSS 2014 (ICL-GNSS 2014)*, June 2014, pp. 1–6.

[204] F. Rottensteiner, J. Trinder, S. Clode, and K. Kubik, "Using the Dempster-Shafer method for the fusion of LIDAR data and multi-spectral images for building detection," *Information Fusion*, vol. 6, no. 4, pp. 283–300, 2005, fusion of Remotely Sensed Data over Urban Areas.

[205] S. Panigrahi, A. Kundu, S. Sural, and A. Majumdar, "Credit card fraud detection: A fusion approach using Dempster-Shafer theory and Bayesian learning," *Information Fusion*, vol. 10, no. 4, pp. 354–363, 2009, special Issue on Information Fusion in Computer Security.

[206] G. Chowdhary and R. Jategaonkar, "Aerodynamic parameter estimation from flight data applying extended and unscented kalman filter," *Aerospace Science and Technology*, vol. 14, no. 2, pp. 106–117, 2010.

[207] Z. Aydogmus and O. Aydogmus, "A comparison of artificial neural network and extended kalman filter based sensorless speed estimation," *Measurement*, vol. 63, pp. 152–158, 2015.

[208] G. Mao, S. Drake, and B. D. O. Anderson, "Design of an extended kalman filter for uav localization," in *2007 Information, Decision and Control*, Feb 2007, pp. 224–229.

[209] C. G. Prevost, A. Desbiens, and E. Gagnon, "Extended kalman filter for state estimation and trajectory prediction of a moving object detected by an unmanned aerial vehicle," in *2007 American Control Conference*, July 2007, pp. 1805–1810.

[210] S.-M. Oh and E. N. Johnson, "Development of uav navigation system based on unscented kalman filter," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, August 2006, pp. 1–21.

[211] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Nov. 2011. [Online]. Available: http://dl.acm.org/citation.cfm?id=1953048.2078195

[212] "Deap: a dataset for emotion analysis using eeg, physiological and video signals," http://www.eecs.qmul.ac.uk/mmv/datasets/deap/, accessed: 2018-10-28.

[213] M. Lichman, "Uci machine learning repository, university of california, irvine, school of information and computer sciences. http://archive.ics.uci.edu/ml."

[214] S. Nittala, C. Elkin, J. Kiker, R. M. J. Curro, K. Xu, V. Devabhaktuni, and A. Reiter, "Pilot skill level and workload prediction for sliding-scale autonomy," in *Proc. of 17th IEEE Int. Conf. Machine Learning and Applications (ICMLA)*, Orlando, Florida, December 2018.

[215] I. Bloch, "Some aspects of Dempster-Shafer evidence theory for classification of multi-modality medical images taking partial volume effect into account," *Pattern Recognition Letters*, vol. 17, no. 8, pp. 905–919, 1996.

[216] B. Yu, K. Sycara, J. Giampapa, and S. Owens, "Uncertain information fusion for force aggregation and classification in airborne sensor networks," in *AAAI-04 Workshop on Sensor Networks*, 2004.

[217] Y. Yang, A. A. Minai, and M. M. Polycarpou, "Evidential map-building approaches for multi-UAV cooperative search," in *Proceedings of the 2005, American Control Conference, 2005.*, June 2005, pp. 116–121.

[218] "MATLAB version 9.4.0," 2018, the MathWorks, Natick, MA, USA.

Appendix A

Source Code

All source code used for this research is released under Apache 2.0 license and can be obtained by contacting the author at colin.elkin@gmail.com.