
SAARLAND UNIVERSITY

Faculty of Mathematics and Computer Science
Department of Computer Science
MASTER THESIS



Digital Pen Features in Machine Learning based Sketch Analysis

submitted by
Alexander Prange
Saarbrücken
February 2021

Advisor:

Prof. Dr.-Ing. Daniel Sonntag
German Research Center for Artificial Intelligence
Saarland Informatics Campus
Saarbrücken, Germany

Reviewer 1: Prof. Dr. Antonio Krüger

Reviewer 2: Prof. Dr.-Ing. Daniel Sonntag

Saarland University
Faculty MI – Mathematics and Computer Science
Department of Computer Science
Campus - Building E1.1
66123 Saarbrücken
Germany

Erklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Statement

I hereby confirm that I have written this thesis on my own and that I have not used any other media or materials than the ones referred to in this thesis

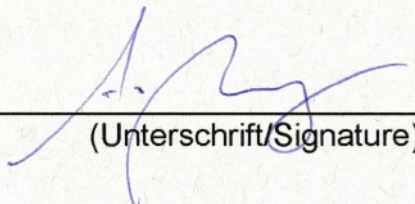
Einverständniserklärung

Ich bin damit einverstanden, dass meine (bestandene) Arbeit in beiden Versionen in die Bibliothek der Informatik aufgenommen und damit veröffentlicht wird.

Declaration of Consent

I agree to make both versions of my thesis (with a passing grade) accessible to the public by having them added to the library of the Computer Science Department.

Saarbrücken, 23.02.2021
(Datum/Date)


(Unterschrift/Signature)

Acknowledgements

I wish to express my thanks to Prof. Lambert Schomaker from the University of Groningen and Prof. Eric Anquetil from IRISA Laboratory, for their support in obtaining a sketch data set, which was believed to be lost.

I want to thank our colleagues Dr. Anika Steinert and Antje Latendorf from the Charité in Berlin for their cooperation and recording of the cognitive assessments data set.

I take this opportunity to thank my colleagues at the German Research Center for Artificial Intelligence (DFKI) for providing a warm, welcoming, and professional work atmosphere. I give thanks to my advisor Prof. Dr.-Ing. Daniel Sonntag, for giving me so many opportunities and for his continuing guidance during my research. I am very thankful to both him and Prof. Dr. Antonio Krüger for allowing me to write this thesis and for their willingness to review it.

Lastly, I want to thank my partner, my family, and my friends for their support.

Abstract

Digital pen features model characteristics of sketches and can be used for various supervised machine learning (ML) applications, such as multi-stroke sketch recognition and behavior characterization. In this thesis, a state-of-the-art set of 165 digital pen features is defined, categorized, implemented, and made publicly available. The feature set is evaluated in the use case of analyzing paper-pencil-based neurocognitive assessments in the medical domain. An interactive cognitive assessment tool is presented that enables physicians to record cognitive assessments using a digital pen. The tool scores the assessments automatically and in real-time, producing structured reports that include visual feedback and transparent explanations about the scoring results. As part of the evaluation it is examined how accurately different feature-based, supervised ML models can automatically score cognitive tests, with and without semantic content analysis. Using standard ML techniques the feature set outperforms all previous approaches on the cognitive tests considered, i.e., the Clock Drawing Test, the Rey-Osterrieth Complex Figure Test, and the Trail Making Test, by automatically scoring cognitive tests with up to 87.5% accuracy in a binary classification task.

To show the generalizability of the digital pen features, the feature set is also compared against the HBF49 ML-based sketch recognition benchmark with the result that by using more features for training, significantly more accurate classification results can be achieved. A series of ML-based sketch recognition experiments is conducted, evaluating ten modern off-the-shelf ML classifiers (i.e., SVMs, Deep Learning, etc.) on a selection of ten sketch data sets from different domains with varying sizes and complexity. In addition, an automated ML approach (AutoML) is explored for fine-tuning and optimizing classification performance on the individual data sets, achieving superior recognition accuracies.

Contents

1	Introduction	1
2	Related Work	3
2.1	Digital Pen Feature Sets	3
2.2	Machine Learning	5
2.3	Cognitive Assessments	8
2.4	Analyzing Cognitive Tests using Digital Pen Features	11
3	Implementation	14
3.1	Digital Ink Library	15
3.2	Digital Pen Features	16
3.2.1	Categorization of Digital Pen Features	16
3.2.2	Syntactic Features	19
3.2.3	Semantic Features	23
3.3	Automated Scoring of the CDT	24
4	Data Collection Method	26
4.1	Handwriting & Sketch Recognition Data Sets	26
4.1.1	CVCsymb Data Set	27
4.1.2	HHReco Data Set	27
4.1.3	ILG Data Set	28
4.1.4	IMISketch Data Set	28
4.1.5	Ironoff-digits Data Set	29
4.1.6	LaViola Data Set	29
4.1.7	NicIcon Data Set	30
4.1.8	Sign Data Set	30
4.1.9	DFKI-symbols Data Set	30
4.1.10	How Humans Sketch Data Set	31
4.2	Cognitive Assessments Data Set	32
4.2.1	Interactive Cognitive Assessment Tool Architecture	32
4.2.2	CDT, TMT & ROCF Data Collection Method	33

5	Evaluation	37
5.1	HBF49 Benchmark Comparison	37
5.1.1	Methodology	38
5.1.2	Results	39
5.1.3	Discussion	39
5.2	Comparison of ML Methods for Sketch Recognition	41
5.2.1	Methodology	41
5.2.2	Results	42
5.2.3	Discussion	43
5.3	AutoML for Sketch Recognition	47
5.3.1	Methodology	47
5.3.2	Results	48
5.3.3	Discussion	48
5.4	ML-based Cognitive Performance Classification	51
5.4.1	Methodology	52
5.4.2	Results	53
5.4.3	Discussion	59
6	Conclusion	61
	Bibliography	63
A	Rubine’s Feature Set	74
B	Willems & Niels Feature Set	77
C	HBF49 Feature Set	90
D	Sonntag et al. Feature Set	97

Chapter 1

Introduction

A multitude of everyday tasks and processes is conducted using pen and paper. These include, among others, note-taking, annotating text, diagramming, sketching, blueprinting, as well as applications in the educational and medical domain. Pen-based writing and sketching stimulate cognition and problem solving across different domains [70]. They can be used to express all types of information, including gestures, language, numbers, symbols, and diagrams. These types of information are easily grasped by humans, but the semantic context and the information encoded in the drawings cannot be easily understood out-of-the-box by a machine. Nowadays, commercially available devices, such as styluses, digital pens and digitizer tablets are capable of recording pen input in real-time. The resulting data is a continuous stream of timestamped x/y coordinates. Various, mathematically defined, geometrical, spatial, temporal, pressure and other features can be extracted directly from the raw input stream. These digital pen features can then be used as input to train supervised machine learning (ML) algorithms for different recognition and classification tasks.

Over the past years, several digital pen feature sets were presented in the scientific literature. Unfortunately, there is a lack of transparently designed, reproducible feature definitions, benchmark experiments and implementations available. This is why the first contribution of this thesis is the definition, categorization and implementation of 165 digital pen features based on related work. The presented set includes state-of-the-art features, and it is the biggest and most comprehensible set of digital pen features yet. In addition, the mathematical formulas for all features are defined in the appendix section of this thesis, and their implementation is made publicly available on GitHub. To further improve the contribution to the scientific community, a software library is published, which includes a standardized structured sketch data exchange format, as well as pre-processing, visualization and format conversion functions. The goal is to provide support for a wide variety of use cases and pen hardware devices, including digital pens on paper, pen-enabled smartphones and tablets, drawing tablets, and even virtual reality applications. The common denominator of these sensor devices is that they provide an output stream of timestamped coordinates, from which the digital pen features can be calculated. In turn, these features can be used to perform ML-based handwriting and sketch analysis in pen-enabled intelligent user interfaces.

The second contribution presented in this work covers a real-world use case from the medical domain. Neurocognitive testing is a noninvasive method for measuring brain function by evaluating specific cognitive abilities, including memory, fine motor control, reasoning, and recognition. Despite recent technological advances, the majority of cognitive assessments used in practice is still conducted using pen and paper with manual scoring by the physician afterwards. This includes verbal interview-like tests, in which the physician takes notes of the patient's answers, as well as tests in which the patient is asked to write or sketch as part of the assessment task. Cognitive assessments have been the subject of recent debate because there are limitations when they are conducted using pen and paper. For example, the collected material is monomodal (written form) and there is no direct digitalization for further and automatic processing. In addition, the results can be biased depending on the physician's level of expertise. Using a digital pen to record neuropsychological tests allows for the analysis of additional parameters that cannot be considered otherwise.

Two approaches for the automatic analysis of paper-pencil-based cognitive assessments are presented here as part of an interactive cognitive assessment tool. The first one is a traditional approach that uses digital pen features to perform a content analysis of drawn sketches and scores the test based on the predefined medical scoring scheme, thereby automating the process that is normally conducted manually by a physician. The next big step in analyzing digital cognitive assessments is to predict cognitive performance independently of the test content, by looking only at the writing and sketching behavior of users, which is explored in the second approach. In this approach the cognitive test performance is predicted by only considering the digital pen features, which are applicable independent of the task, without performing further content analysis. The aim is to support more automatic, more objective and accurate diagnostics of pen sensor input, which can be used in hospitals and retirement homes to transparently evaluate cognitive performance (i.e., without explicit testing), to guide medical interventions, and to adapt cognitive training in a personalized manner.

One of the fundamental tasks in pen-based user interfaces is sketch recognition, which is concerned with the automatic labeling of sketches with a certain class, e.g., circle, rectangle, text, airplane, clock etc. A common approach is to train feature-based supervised ML models for the classification of sketches. The third contribution of this thesis is the comparison of the presented set of 165 digital pen features to a public benchmark that uses only 49 features, to test the hypothesis that more features allow for better sketch recognition performance. However, one limitation of this benchmark is that it only considers two ML algorithms, which is why the fourth contribution of this work is a comparison of ten modern off-the-shelf ML classifiers on a selection of ten sketch data sets from different domains. The aim is to guide researchers in the selection of ML algorithms for sketch recognition, by testing the classification performance on a variety of sketches of varying complexity, which were recorded using different pen interfaces. In addition, an automated ML approach (AutoML) is evaluated, to show how the feature-based classification approach can achieve superior recognition performance when it is fine-tuned and optimized on individual data sets.

The rest of this thesis is structured as follows. Chapter 2 covers related publications from the field of digital pen features, ML and cognitive assessments. Chapter 3 describes the implementation details, including the definition of digital pen features and their categorization. In chapter 4 the architecture of the interactive cognitive assessment tool and the method for collecting the sketch data sets are detailed, and a summary of all data sets is presented. The evaluation is presented in chapter 5, followed by a summary and discussion of future work in chapter 6.

Chapter 2

Related Work

This chapter summarizes relevant related publications in four sections. Section *Digital Pen Feature Sets* summarizes the features considered in this thesis, section *Machine Learning* provides an overview of different ML methods, section *Cognitive Assessments* covers neurocognitive testing instruments, and the last section focuses on *Analyzing Cognitive Tests using Digital Pen Features*.

For the remainder of this work series of timestamped x/y coordinates are referred to as *strokes*, and a collection of continuous, interrelated strokes is referred to as a *sketch*. In this context gestures are sketches conveying a particular meaning, e.g., arrows or text strikethroughs. The terms *cognitive assessments* and *cognitive tests* are interchangeable.

2.1 Digital Pen Feature Sets

Traditionally, stroke level features are most often used for statistical gesture recognition. One of the most prominent sets of pen features was presented by Dean Rubine in 1991 [88]. It contains a total of 13 features that are designed to reflect the visual appearance of strokes in order to be used in a gesture recognizer. The feature set includes geometric features, such as cosine and sine values of the initial angle of the gesture, the length of the gesture or total angle traversed, and temporal aspects, such as maximum speed and duration of gestures. According to the author, these features were determined empirically to work well on several different gesture sets. Applying linear classifiers to these gesture sets, Rubine reports recognition rates of over 96%, even for relatively small training set sizes of 15 samples per class. He also presents GRANDMA (Gesture Recognizers Automated in a Novel Direct Manipulation Architecture), a toolkit that allows users to define new gestures by example, thereby eliminating the need to hard-code recognizers for different gesture sets. However, one major limitation is that all features are defined on single-stroke gestures exclusively. Rubine states, that this is an intentional limitation of GRANDMA, and a contrast to multi-stroke gesture-based systems by design [88]. By limiting recognition to single-stroke gestures, the segmentation problem of multi-stroke recognition is avoided [99], enabling shorter timeouts and response times in interactive interfaces. According to Rubine, single-stroke gestures represent a single intentional

physical user action, a property thought to contribute positively to the usability of user interfaces [14]. Nevertheless, Rubine’s work presents one of the first and most cited, generalized and reproducible feature sets available for statistical gesture recognition. It has been successfully applied in pen-based intelligent user interfaces [98, 112], multi-touch gesture recognition [19, 85] and even eye-tracking analysis [2, 3].

More recent work by Don J.M. Willems and Ralph Niels [108] defines a total of 89 features using formal mathematical descriptions and algorithms. While their technical report mainly focuses on the implementation details, their other publications show ML-based applications of their feature set for multi-stroke gesture recognition [109, 110, 111], forensic writer identification [67] and writer verification [13]. The Willems & Niels feature set includes several features, which are also present in Rubine’s set [88], but it is overall comprised of computationally more complex features. These include curvature, perpendicularity, complex hull properties, histogram analysis, acceleration and many more. It is also one of the few feature sets that includes force-based characteristics. These are features related to the pressure that users apply to the writing surface, and which, depending on the hardware device, is either measured directly by the digital pen or the digitizer tablet. In addition, the Willems & Niels feature set contains higher level *meta features*, which model crossings, connected strokes and straight lines. Such features are beneficial for the classification of multi-stroke gesture recognizers. Overall, this feature set poses a relevant source for the modeling of sketch characteristics, which can be easily reproduced and applied to a variety of use cases and ML algorithms.

In 2013, Adrien Delaye and Eric Anquetil introduced the HBF49 (Heterogeneous Baseline Feature) set [25], which contains 49 features and is specifically designed to be used as a reference for the evaluation of symbol recognition systems. Similar to Rubine [88] an empirical constructive approach is adopted for designing this feature set, with the aim to handle a large diversity of symbols in various experimental contexts [25]. In contrast to Willems & Niels [108], Delaye & Anquetil only consider the simplest features from each feature category (geometrical, temporal, etc.), to maintain a feature space of limited dimension [25]. They evaluate their 49 features using a standard SVM and a simple 1-Nearest-Neighbor classifier on eight data sets with considerable diversity of content (digits, characters, symbols, geometrical shapes and gestures). Results show that using off-the-shelf statistical classifiers, the HBF49 representation performs comparably or better than state-of-the-art results reported on these hand-drawn objects (>90% accuracy using SVMs). The transparency of the approach, the details of the feature design, and the presentation of their experimental protocol make it possible to use the HBF49 feature set as a benchmark for comparing the classification performance of different feature sets. After overcoming several obstacles, it was possible to obtain all of the eight data sets required to reproduce the presented experiments in this thesis.

Another set of 14 features described by Sonntag et al. [96], presented as part of a pen-based interactive decision support system for radiologists, is included in the selection. A common practice in hospitals is that a radiologist’s dictated or written patient report is transcribed by hospital staff and sent back to the radiologist for approval, which takes a lot of time and lacks direct digitization of pen input. Sonntag et al. present a system, which allows doctors to use a digital pen to fill out the paper-based structured reporting form, with direct digitization of pen input in real-time. The input is not limited to numbers and text, but can also include hand-drawn sketches, free text annotations and correction gestures. Instead of forcing the user to switch manually between writing, drawing, and gesture mode, a mode-detection system is deployed to predict the user’s intention based on the sketch input. To classify the input, a number of features are calculated, including compactness, eccentricity, closure and others. These features are used in

a multi-classification and voting system to detect the classes of handwritten information, shape drawings, or pen gestures [96]. According to Sonntag et al., the system reaches a recognition rate of nearly 98%. The presented work shows how digital pen features can be used transparently in pen-based intelligent user interfaces to improve interactivity and reduce cognitive load for the user. In addition, the reproducible description of the features makes the feature set an ideal candidate to be considered in this work.

So far 165 digital pen features from the four aforementioned feature sets [25, 88, 96, 108] are included in the selection. In addition, 11 features are considered, collected from literature focused on the evaluation of cognitive assessments [21, 24, 107]. These features include the number of strokes, sketching time, stroke distance, duration, average pressure, average velocity, the variation of velocity, number of pauses, average pause duration, the ratio between sketching and pausing, and average lift duration. A comprehensible summary of these publications is given in section 2.4 (*Analyzing Cognitive Tests using Digital Pen Features*). By directly analyzing the characteristics of pen input and sketching behavior, a causal link between sketch characteristics and cognitive test performance is created, without the need to analyze the sketch content itself. This way cognitive behavior is classified transparently during user interaction and independent of the task at hand, thereby enabling additional opportunities in interactive systems, such as evaluation, feedback and content adaptation of pen-based interfaces in real-time.

In summary, a total of 165 (+11) digital pen features are considered, which are computable in real-time, and, among others, cover geometrical, spatial, temporal, and pressure properties of the sketch.

2.2 Machine Learning

This thesis is concerned with supervised machine learning methods, which cover the ML task of learning models that map an input to an output based on examples of input-output pairs [89]. The input is typically a set of labeled data in form of a feature vector, on which the ML algorithm is trained to classify unseen samples by predicting their labels [63]. In the context of ML, a feature describes a measurable property or characteristic of a data sample [9] and a label describes the category assigned to a sample. Using the above described feature sets, a vector of features can be calculated for each sketch directly from the digital pen input. All use cases in this thesis are concerned with classification problems, where labels are either binary or multi-class categories.

The publications presented in this chapter cover a wide selection of supervised ML algorithms, which are explained in more detail in the remainder of this section. One of the basic pattern recognition algorithms, the *k-Nearest-Neighbors* algorithm, was introduced by Fix & Hodges [100]. This non-parametric classification algorithm uses distance metrics (e.g., Euclidean distance) to calculate closest neighboring samples in the feature space. During the training phase, the algorithm only stores the feature vectors and class labels, whereas in the classification phase the unlabeled sample is assigned the label which is most frequent among the *k* training samples nearest to it [89, 100].

Despite its name, *Logistic Regression* is a linear model used for classification rather than regression, which is a very popular technique for dealing with binary classification problems [9]. Logistic regression is a statistical model that uses a logistic function to model the probability of the input belonging to a certain class. Most commonly the output is transformed using the logistic sigmoid function to return a probability value which can then be mapped to two or more discrete classes [89]. *Naïve Bayes* describes a family of

probabilistic classifiers based on the Bayes' theorem, which covers the "naive" assumption of conditional independence between every pair of features given the value of the class variable [63, 89]. It is called naive, because the conditional independence assumption on which it is based, is rarely true in real-world applications [115]. Nevertheless, Naive Bayes classifiers deliver excellent performance for many complex real-world applications and are highly scalable while requiring only a small amount of training data to estimate their parameters [115]. However, for some problems, other approaches, such as *Boosted Decision Trees* or *Random Forests* outperform Naive Bayes classifiers, as shown by Caruana & Niculescu-Mizil [17]. *Decision Trees* are non-parametric tree-like graphs, created with the goal to predict the label by learning simple decision rules inferred from the features [113]. The leaves of the graph are the class labels and the inner nodes represent if-then-else decision rules. A major benefit of decision trees is the inherent transparency of the resulting model, as the trees can be easily visualized, and any given situation is observable and can be easily explained by Boolean logic. On the other hand, the trees can become very complex, hence not generalizing the data well (overfitting), and learning an optimal decision tree is known to be NP-complete even for simple settings [48, 51]. This is one of the reasons why ensemble methods, such as *Boosted Decision Trees* create more than one decision tree, based on previously mislabeled training samples, which can improve performance in certain settings [51]. Similarly, *Random Forests* construct a number of decision trees for various sub-samples of the dataset and averaging over them to improve accuracy and compensate for the overfitting problem [42]. Another version of decision tree is the *Gradient Boosted Decision Tree*, which uses boosting to optimize weaker decision trees [12]. In supervised machine learning, boosting describes an ensemble method that involves training multiple models (decision trees) in sequence, where the error function used to train each model depends on the performance of the previous models [9]. Boosting is also used in other ML methods, such as the *AdaBoost* (adaptive boosting) ensemble algorithm, which begins by fitting a classifier on the original data set and then fits additional copies of the classifier on the same data set [9]. For each copy, the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus on misclassified samples [9].

At the end of the last century *Support Vector Machines* (SVMs) became quite popular for solving classification problems in high dimensional spaces [9]. They do so by finding a hyperplane (decision boundary) in a high dimensional space that distinctly classifies the sample points [22, 89]. Data points falling on either side of the hyperplane can be attributed to different classes. There is a direct correlation between the number of features used as input and the dimension of the hyperplane [22]. SVMs can perform a non-linear classification by implicitly mapping their inputs into high-dimensional feature spaces, which is called kernel trick [9]. In the scope of this thesis, two different types of SVM-kernels are considered, the linear and the radial basis function kernel (RBF), which is especially useful when the data points are not linearly separable [22].

Another family of ML methods, the so-called artificial neural networks, become increasingly popular in the last few years [37]. The current wave of popularity began with a breakthrough by Hinton [45], who showed that a specific type of neural network, called *deep belief network*, can be trained efficiently. Bengio et al. [7] and Ranzato et al. [81] showed shortly afterwards that this method can be extended to other neural networks. At this time the term *Deep Learning* was coined, which emphasizes that researchers are now able to train larger neural networks (deeper networks with many layers) than ever before [37]. Although many of the algorithms and concepts are known for decades, it were the breakthroughs in computing capacities in recent years that led to practical applications that can work efficiently on large data sets [37]. Especially the development

of high-performance graphics processing units (GPUs) enables a growing complexity and performance of neural networks [26]. *Convolutional Neural Networks* (CNNs) are one type of network commonly used for image recognition, but they can also be used for sketch classification from digital pen input [52, 116]. A novel approach by Moetesum et al. [62] investigates how CNNs and more traditional ML methods (e.g., Random Forests, Naive Bayes, SVMs, etc.) can be combined to analyze deformations in neuropsychological drawings. Unfortunately, their results are not directly transferable for the scope of this thesis, as they focus on the images of sketches rather than the pen input.

In contrast, *Long Short-Term Memory* (LSTM) approaches have proven to be successful in classifying handwriting input directly from the continuous digital pen signal [16, 43]. Similarly, *Recurrent Neural Networks* (RNNs) deliver state-of-the-art performance for sketch recognition, e.g., the Sketch-RNN by Ha & Eck [40], which is part of Google Quick Draw¹. Interestingly enough, Sketch-RNN uses as sequence-to-sequence *Variational Auto-Encoder* (VAE) to automatically generate latent feature vectors from the sketch input, which are then analyzed by the RNN. Another example of RNNs, the SketchSegNet+ by Qi & Tan [80] was trained on a subset of the Quick Draw data set to perform multi-class sketch segmentation. Essentially, they take stroke sequence order as well as the contextual information among strokes into account to determine the semantic labels of input strokes. RNNs and LSTMs can also be used for signature verification [58] and to generate handwriting samples from textual input [10, 39], which can be used for interactive ink editing in HCI, such as spell-checking and correction of handwriting [1].

However, the major drawback of DL models is their inherent blackbox approach, which lacks transparency and explainability about the recognition process. More classic ML approaches often deliver a more easily understood classification. This is important in the medical use cases discussed in this thesis, as physicians need to understand the predictions in order to trust the output of the model. Another issue is the vast amount of data that is required to train DL models, e.g., the Google Quick Draw data set contains about 50M drawings from 15M players. The concept of transfer learning in the context of sketch recognition has been proposed only recently [62, 92], and still requires a lot of data. Despite these limitations, the evaluation in chapter 5 also covers DL models.

The majority of the presented ML/DL methods consists of parameterized algorithms. These so-called *hyperparameters* are used to control the learning process since the same kind of ML model can require different constraints, weights or learning rates to generalize on different data sets [30]. In ML, *Hyperparameter Optimization* (HPO) is the task of choosing a set of optimal hyperparameters for a learning algorithm [9]. The main task of automated machine learning (AutoML) is to automatically perform HPO to enhance performance and reduce the human effort necessary for applying machine learning [30]. Several studies have shown that AutoML approaches improve the performance of classic ML algorithms by tailoring them to the problem at hand [30, 59, 93]. In addition, this approach can improve the reproducibility and fairness of scientific studies, since it applies the same level of tuning for all methods and therefore provides fairer comparisons [8, 30]. Several frameworks are nowadays available for AutoML, such as auto-sklearn [31, 32], which is a drop-in replacement for the famous scikit-learn library², the open-source H2O platform used by RapidMiner [61] and Google AutoML Tables³, which is utilized for HPO in this thesis.

¹Quick Draw is an online game developed by Google that challenges players to draw a picture of an object or idea and then uses a neural network to guess what the drawings represent. The game is publicly available at <https://quickdraw.withgoogle.com/>.

²<https://scikit-learn.org/>

³<https://cloud.google.com/automl-tables>

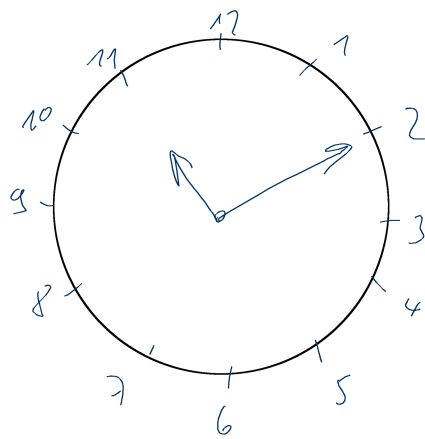
Assessment	Execution Time	Pen Input	Sketch Shapes
AKT [35]	15 min	100%	Cross-out
CDT [34]	2-5 min	100%	Clock, digits, lines
CERAD [64]	30-45 min	20%	Pentagons, rectangles, circle, ...
DemTect [53]	6-8 min	20%	Numbers, words
MMSE [33]	5-10 min	9%	Pentagons
MoCA [65]	10 min	17%	Clock, digits, lines
ROCF [15]	15 min	100%	Circles, rectangles, triangles, ...
TMT [83]	3-5 min	100%	Lines

Table 2.1: Comparison of widely used cognitive assessments including the percentage of tasks with pen input that contribute to the total score. Execution time does not include the time needed by the physician to score the test (several minutes, depending on test).

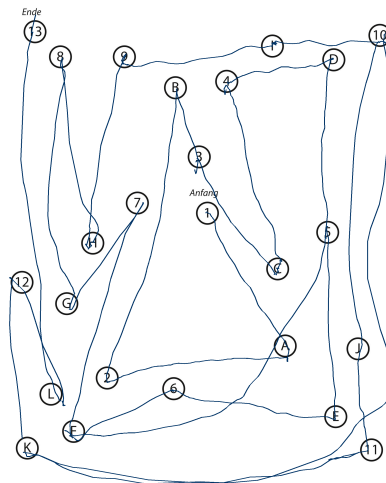
2.3 Cognitive Assessments

Neurocognitive testing is a noninvasive method for measuring brain function by evaluating specific cognitive abilities, including memory, fine motor control, reasoning and recognition. Despite recent technological advances, the majority of cognitive assessments used in practice is still conducted using pen and paper with manual scoring by the physician afterwards. This includes verbal interview like tests, in which the physician takes notes of the patient’s answers, as well as tests in which the patient is asked to write or sketch. In this work, a selection of paper-pencil cognitive assessments is considered based on feedback from domain experts and a recent market analysis of existing, widely used, cognitive assessments conducted by Niemann et al. [69]. These assessments were successfully digitalized during the Interakt project (Interactive Cognitive Assessment Tool) [94], and are summarized in table 2.1, namely Age-Concentration (AKT) [35], Clock Drawing Test (CDT) [34], CERAD Neuropsychological Battery [64], Dementia Detection (DemTect) [53], Mini-Mental State Examination (MMSE) [33], Montreal Cognitive Assessment (MoCA) [65], Rey-Osterrieth Complex Figure (ROCF) [28], and Trail Making Test (TMT) [83]. This selection of tests accounts for a variety of patient populations and test contexts. In this thesis, the focus is on the CDT, the TMT and the ROCF, as they have the highest ratio of pen input relevant to the scoring result. The AKT is discarded, because it only consists of the rather simple task of crossing out figures, and preliminary testing results show that the samples have too few strokes for analysis.

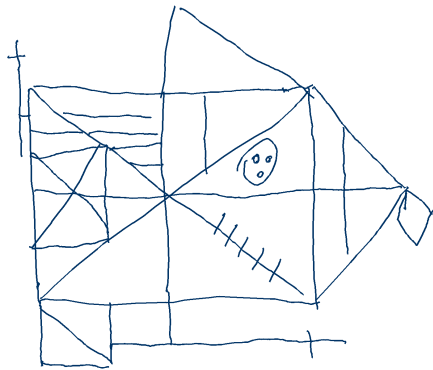
For more than 50 years, the CDT is used as an assessment tool for cognitive impairment. It is a simple paper and pencil test in which the participant is asked to draw a clock face and indicate a certain time (see figure 2.1a). The task is primarily designed to test the visuospatial ability and is often used in geriatrics to screen for signs of dementia, such as Alzheimer’s disease, or other neurological conditions, including Parkinson’s disease, traumatic brain injury, and stroke recovery. Usually a trained professional observes the clock drawing task and scores the final sketch based on a scoring scheme, which takes up to a few minutes. Automatically scoring the CDT has several benefits: Firstly, it significantly reduces the time caregivers have to invest into administering the CDT; and secondly, it is likely to produce more objective scores and potentially enables a more detailed analysis [94]. Not all scoring schemes are equally well suited for automation, since most of them have been designed to be quick and easy to be interpreted by human testers. The 20-point Clock Drawing Interpretation Scale (CDIS) by Mendez et al. [60] is selected in this thesis, because it contains clear test parameters that can be modeled



(a) Clock Drawing Test.



(b) Trail Making Test B.



(c) Rey-Osterrieth Complex Figure Test.

Figure 2.1: Samples of cognitive assessments recorded during the Interakt project.

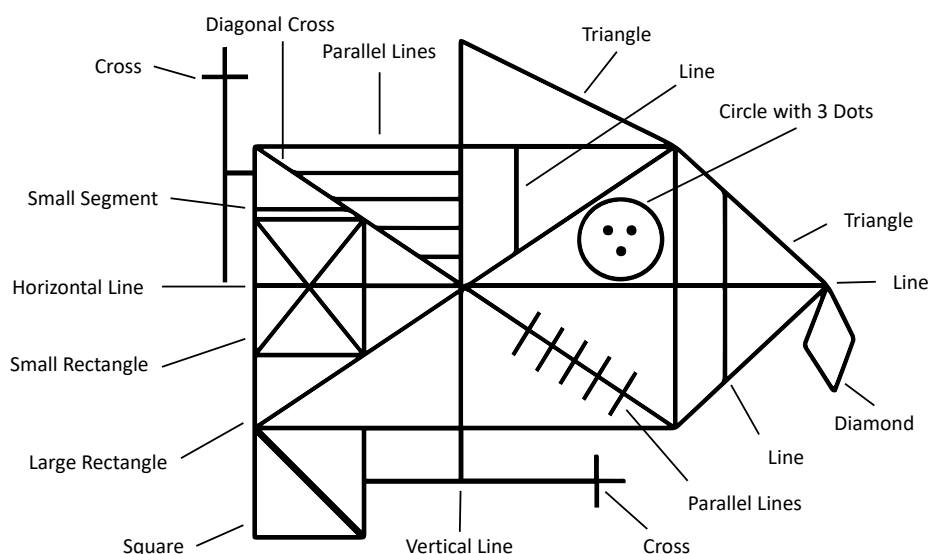


Figure 2.2: The Rey-Osterrieth Complex Figure (ROCF) is composed of 18 sub-figures.

mathematically and computationally. In addition, the manual scoring procedure of CDIS is very time-consuming and would highly benefit from automatic computation. The CDIS contains items such as "All numbers 1-12 are present", which are to be rated 0 if not fulfilled and 1 if satisfied. All 20 individual scores are then added up and the final score indicates the severity of cognitive impairment. For example, a score of 18 or less is likely to indicate Alzheimer's and similar forms of dementia.

Frequently used in neuropsychological testing, the Trail Making Test (TMT) [83] is a standardized paper and pencil test in which the participant is asked to connect numbered nodes similar to a child's connect-the-dots puzzle (see figure 2.1b). In its original form, it was part of the Army Individual Test Battery (1944) and was later subsequently incorporated into several cognitive test batteries [102]. The TMT is widely available and can be easily administered in practice [23]. In addition, it can be used to assess a variety of neurological disorders [90]. Here, the use case of geriatrics is considered, where TMT versions A and B are used to screen for signs of dementia. Each of the two parts (A and B) consists of 25 encircled items on an A4 sheet of paper and subjects are asked to draw a line through them in the correct order as quickly as possible without lifting the pen. TMT-A involves number sequencing (1 to 15), whereas TMT-B includes set-shifting: it requires the subject to alternate between numerical and alphabetic sequences (1-A-2-B-3...). The main performance indicator used in clinical practice is the total completion time, which is manually measured by the physician using a stopwatch [11, 57]. The total scoring of the cognitive assessment is then calculated by comparing the completion time to age-specific normative values [23, 57].

An even more complex example of a neuropsychological assessment, that is rated entirely based on pen input, is the ROCF [15, 28]. A printed Rey-Osterrieth figure template is presented to the subject, who is then asked to copy it onto a blank piece of paper (see figure 2.1c). Then the physician hides the template and the subject is asked to immediately recall the figure from memory and sketch it onto another blank piece of paper. This is repeated once again after approximately 30 minutes (delayed recall). Afterwards the physician scores each sketch based on the visual appearance of the 18 sub-figures of the

ROCF (see figure 2.2). Each sub-figure is rated between 0 (non existent) and 2 (drawn and placed correctly), and the individual scores are summed up to produce the final score. In contrast to the CDT, the TMT and ROCF have only one scoring scheme that is commonly applied in practice according to medical guidelines.

2.4 Analyzing Cognitive Tests using Digital Pen Features

Cognitive assessments have been the subject of recent debate because there are limitations when they are conducted using pen and paper. For example, the collected material is monomodal (written form) and there is no direct digitalization for further and automatic processing. Additionally, the results can be biased depending on the physician's level of expertise. To mitigate these shortcomings, digitizing and analyzing paper-and-pencil assessments has been introduced recently [95]. Using a digital pen to record neuropsychological tests allows for the analysis of additional parameters that cannot be considered otherwise. In Heimann-Steinert et al. [44] we show that replacing a standard ballpoint pen with a digital pen has no influence on neurocognitive test results. Werner et al. [107] perform an experiment in which they record common cognitive assessment tasks using a digitizer tablet to compare the handwriting behavior of healthy controls to patients suffering from mild cognitive impairment (MCI) and Alzheimer's disease. Their feature set includes on paper time, in air time, the ratio between in air and on paper time, on paper length, velocity and pressure. They report significant differences between the groups in almost all measures [107], for example, temporal measures are higher and pressure is lower in the cognitively impaired groups. Using ANOVA, Werner et al. classify 69% to 72% of the participants correctly, although the classification for the MCI group is reported to be relatively poor. Findings by Schroter et al. [91] support the statement that it is possible to distinguish between different forms of cognitive impairment and healthy subjects by analyzing handwriting movements. They find that both patients with MCI and patients with probable Alzheimer's exhibit loss of fine motor performance and that the movements of Alzheimer patients are significantly less regular than those of healthy controls. In both publications the classification is made entirely without content analysis, solely based on the handwriting behavior of subjects.

Similar findings are reported by Davis et al. [24], who use an off-the-shelf digitizing ballpoint pen to analyze the cognitive abilities of patients with dementia, Alzheimer's and Parkinson's disease, by deploying a digital version of the CDT. A set of digital pen features, including temporal aspects, such as speed and pauses, is used as input for a collection of ML algorithms, including SVMs, random forests, boosted decision trees, and others. They report a top accuracy of 82% using SVMs to classify dementia, but it is not clear which features exactly are used. However, Davis et al. report that time-dependent variables prove to be important for detecting cognitive change. They can reveal when individuals are working harder, even though they are producing normal-appearing outputs, e.g., the total time to draw the clock differentiates those with amnesic mild cognitive impairment (aMCI) and Alzheimer's disease from healthy controls [24]. In Cohen et al. [21] the same authors elaborate on these findings by comparing the ratio of drawing versus non-drawing time for older adults with depression during the CDT. Their evaluation reveals a significant effect of age on drawing times during challenging tasks, while there is no significant effect during simpler tasks like copying figures. This could prove highly relevant for interactive pen-based systems, in which content and task difficulty need to be adjusted in real-time, depending on the cognitive load of users. Oviatt et al. report similar findings in the educational domain, where they use digital

pen features to predict task expertise and user performance in mathematics [71, 72, 117]. Using SVMs, Random Forests, and Naive Bayes classifiers, they achieve up to 92% prediction accuracy with features, such as average number of pen strokes, total writing time, stroke distance, duration, pressure, and speed. Similar techniques have been used to diagnose other cognitive impairments, such as Parkinson’s disease [27, 55], or to predict task difficulty and user performance of cognitive tests [6]. This shows that the ML-based digital pen features approach considered in this work can be applied to a variety of application domains and use cases.

Digital pen features are also useful to perform a traditional content analysis of cognitive assessments. Automatic scoring systems mimic the scoring procedure performed by physicians, which is regulated by medical guidelines and usually produces a numeric score. In Prange et al. [79] we show how to model cognitive status through automatic scoring of a digital version of the CDT. We implement the Mendez scoring scheme and create a hierarchy of error categories that model the test characteristics of the CDT, based on a set of impaired clock examples provided by a geriatrics clinic. Using a digital pen we record 120 clock samples for evaluating the automatic scoring system, with a total of 2400 samples distributed over the 20 scoring classes of the Mendez scoring scheme. Samples are scored automatically using a handwriting and gesture recognition framework based on digital pen features and results show that we provide a clinically relevant cognitive model for each subject. In addition, we heavily reduce the time spent on manual scoring. A similar automated analysis of the CDT based on ML is presented by Souillard-Mandar et al. [97]. They design and compute a large collection of content related features and explore the performance of classifiers built using a number of different subsets of these features and a variety of ML techniques. The authors use traditional ML methods to build prediction models that achieve high accuracy in detecting cognitive impairment. However, their major drawback is, that the deployed feature set is not reproducible, nor is there a data set available for repeating the experiments.

Another very popular assessment is the Mini-Mental State Examination (MMSE) introduced in 1975 by Folstein et al. [33]. The MMSE is a 30-point questionnaire, which is administered by a trained professional, who leads the subject through the questionnaire and sketching tasks whilst taking notes. The CDT is often administered together with the MMSE, since both assess different, complementing cognitive abilities [74]. In Prange et al. [77] we present a multimodal system for the automatic execution and evaluation of the MMSE, that uses speech analysis in combination with handwriting and gesture recognition based on digital pen features. Taking into account multiple sensor inputs is the next step to improve neurocognitive testing by taking advantage of multimodal, multi-sensor interfaces [73]. In Niemann et al. [69] we describe the architecture of a multimodal multi-sensor assessment framework for cognitive impairment that combines digital pen strokes, pen sensors, automatic handwriting and gesture recognition and electrodermal activity (EDA). We use EDA for potential inclusion of open sympathetic arousal, stress and affect (e.g., emotion detection through autonomic nervous system signals). To include biosensors (e.g., EDA) into future digital pen-based interfaces is very interesting because it is a process-tracing method (unobtrusive and continuous measure) for neural activity and can reflect psychological processes. The digital pen-based environment provides a sensor fusion context for its interpretation. In the future, large scale community screening programs can arise from resulting multimodal data collections to identify multimodal profiles of impairment across different cognitive, psychiatric and functional domains/abilities. It will also help to guide differential diagnosis and further neurocognitive assessment, especially because multimodal digital assessments are unbiased to a large degree.

Clinically relevant examples of digitalized pen-based cognitive assessments also include the ROCF [15, 28], which can be used for various purposes, such as diagnosing the periphery [20]. The authors show that accurate reproduction of spatial features predictably declined as the target was presented further in the visual periphery. Such analysis of the periphery in the context of the ROCF drawing tasks has shown relevant for a variety of medical conditions, including age-related macular degeneration and apoplectic stroke [20]. Wang et al. [106] report a significant correlation between ROCF measures, tremor and bradykinesia (impaired and slow body movement), such as is often present in patients suffering from cognitive impairment (i.e., Parkinson’s disease). Similarly, Salthouse [90] shows that the TMT can be used to assess a variety of neurological disorders and it can even be adopted for diagnosing neurological disorders in children [82]. His findings are backed by Barz et al. [6], who use digital pen features to predict task difficulty and user performance of the TMT. Their automated evaluation uses a subset of the features considered in this thesis and shows that a correlation-based feature selection can be beneficial for ML-based model training in certain scenarios. Similar results were reported by Dahmen et al. [23], who investigated the utility of features in analyzing a digital version of the TMT. Their considered features include, among others, the average pause duration, number of lifts and average pressure.

To summarize, utilizing digital pen features, in ML-based automated cognitive assessment systems, opens up an opportunity to analyze handwriting and sketching behavior, that could not be considered otherwise. Related work shows that these features can be used for both, sketch content analysis and the unobtrusive, transparent modeling of cognitive behavior from raw digital pen input. Furthermore, this thesis is motivated by the lack of a comprehensible and publicly available feature set for such purposes.

Chapter 3

Implementation

Motivated by the lack of a publicly available feature collection, this section details one major contribution of this thesis, namely the categorization and implementation of 165 digital pen features, including their public release on GitHub⁴. In addition, the *Digital Ink Library* is presented, which specifies a structured stroke and sketch data format, including conversion tools, and allows users to calculate digital pen features on their data. Of course, the library, too, is available on GitHub in several programming languages.

Over the past few years, the availability of digital pen hardware has increased drastically and there is a wide variety of devices to choose from, which produce sketch samples in varying data formats. In this work, the focus lies on the similarities between the most commonly used hardware devices for pen-based user interfaces. As not all technologies deliver the same type of sensor data, a subset is identified that is covered by the majority of input devices. It is referred to as digital ink, a set of time-series data containing coordinates and pressure at each timestamp. For the remainder of this paper the following notation is used:

$$x, y : \text{coordinates} \quad (3.1)$$

$$p : \text{pressures} \quad (3.2)$$

$$t : \text{timestamps} \quad (3.3)$$

A series S of n sample points between a pen-down and pen-up event is called a stroke and can be represented as a series of tuples:

$$S = (x_0, y_0, p_0, t_0), (x_1, y_1, p_1, t_1), \dots, (x_{n-1}, y_{n-1}, p_{n-1}, t_{n-1}) \quad (3.4)$$

Where x_i represents the x coordinate of the i-th sample point within the series with $0 \leq i < n$. The tuple itself is referenced by s_i . Timestamps are measured in milliseconds and it is unimportant if they are absolute or relative to the first point.

⁴<https://github.com/DFKI-Interactive-Machine-Learning>

3.1 Digital Ink Library

During the collection of sketch data sets it became evident that there is no standard data format for digital ink samples currently in use. Hence, as part of this thesis, the *digital ink library* is implemented to provide a simple and extensible standardized data exchange format with conversion tools for common formats. Currently, the API is available in several programming languages, including Python, Rust and Java. The JSON (JavaScript Object Notation) open standard file format is used to encode all necessary information of strokes, such as x/y coordinates, timestamps and pressure (see listing 3.1). This format is highly extensible, as additional information can be easily stored in the *meta* field of each stroke. Depending on the use case this can include strings, such as class labels, array-like information, such as point-wise annotations or any other kind of serializable data object.

Listing 3.1: Structured JSON data exchange format for strokes.

```

1 {
2   "type": "stroke",
3   "meta": {
4     "<key1: string>": <value1>,
5     "<key2: string>": <value1>,
6     ...
7   },
8   "x": [<x1: float>, <x2: float>, ..., <xn: float>],
9   "y": [<y1: float>, <y2: float>, ..., <yn: float>],
10  "timestamp": [<t1: long>, <t2: long>, ..., <tn: long>],
11  "pressure": [<p1: float>, <p2: float>, ..., <pn: float>]
12 }
```

Similarly, sketches are encoded as a collection of strokes with the option to include additional meta information (see listing 3.2). The digital ink library also includes several conversion functions, i.e., scaling and normalization, which help prepare sketch data samples as input for ML experiments. A collection of visualization tools is provided to render sketches as images and videos. Having a real-time representation of the sketch as it was originally drawn is a major benefit of digital ink data, as it enables an analysis of the sketching process at a later time. This is especially useful in the use case of cognitive assessments, as the behavior of patients can be analyzed over time. In addition, the order of strokes is not evident from a static image, but can easily be retraced by watching the playback of the video.

Listing 3.2: Structured JSON data exchange format for sketches.

```

1 {
2   "type": "sketch",
3   "meta": {
4     "<key1: string>": <value1>,
5     "<key2: string>": <value2>,
6     ...
7   },
8   "strokes": [<stroke1>, <stroke2>, ..., <stroken>]
9 }
```

3.2 Digital Pen Features

Individual, measurable properties or characteristics of digital ink are referred to as features. Features are calculated directly from the input sample points and represented by a numerical value. Therefore, a feature can be seen as a mapping function:

$$f : S \mapsto \mathbb{R} \quad (3.5)$$

Depending on the feature, S can be a set of strokes (gesture level), a single stroke (stroke level) or a subset of sample points. Most commonly a vector of features F is calculated from the input data, which can then be used in ML-based classifiers:

$$F = [f_1(S), f_2(S), \dots, f_m(S)] \quad (3.6)$$

Unfortunately, there are currently no comprehensible features sets available that could be used off-the-shelf by researchers. This is why in this thesis a definition, categorization and implementation of 165 digital pen features is presented. The collection of features is based on the four feature sets discussed in section 2.1 (*Digital Pen Feature Sets*), while a fifth set of features, related to cognitive assessments, is presented as part of the experiments in chapter 5 (*Evaluation*). The remainder of this section provides a categorization and brief overview of the features, while their mathematical definitions are detailed in the appendix (A: *Rubine's Feature Set*, B: *Willems & Niels Feature Set*, C: *HB49 Feature Set*, and D: *Sonntag et al. Feature Set*).

As the majority of today's ML frameworks and libraries are available in the Python programming language, the implementation, too, is done in Python 3.7. It is designed in a way that enables users to either calculate individual features or entire vectors at once, assuming that the input is provided in the digital ink library format. During preliminary experiments, it turned out that calculating all features on data sets of more than 10,000 samples can take quite some processing time on common computer hardware. Because this possibly presents a major limitation for bigger data sets common in ML research, an alternative version written in the Rust programming language was implemented. Rust is a relatively new programming language introduced by Mozilla Research in 2010, which is designed for performance and safety, especially in concurrent contexts [54]. In contrast to Python, an interpreted, high-level and general-purpose programming language, Rust code is eventually compiled to native machine code and therefore delivers performance comparable to C++ code, but with guaranteed memory safety. As a result, the Rust implementation is able to calculate digital pen features up to ten times as fast as the Python reference API. In addition, Rust code can be deployed on multiple platforms, including iOS and Android, making it a perfect candidate for mobile interactive pen-based applications.

3.2.1 Categorization of Digital Pen Features

Due to the nature of sketched and handwritten input, there are a few features and concepts that the considered feature sets have in common. Before going into detail about the categorization, it is useful to gain a general understanding of digital pen features. The most prominent example of a common feature is the length of a stroke, where the Euclidean distance is used to measure the gap between sampling points. Given two

sampling points $q = (x_q, y_q)$ and $r = (x_r, y_r)$ their distance is calculated as follows:

$$\|qr\| = \|r - q\| = \sqrt{(x_r - x_q)^2 + (y_r - y_q)^2} \quad (3.7)$$

The length of a stroke (a sequence of sampling points) is given by the sum of distances between the sampling points:

$$f_{StrokeLength} = \sum_{i=1}^{n-1} \|s_i - s_{i-1}\| \quad (3.8)$$

A bounding box (see figure 3.1) around a set of strokes describes the smallest enclosing rectangular area containing the entire set of points. Its size is determined by the minimum and maximum sample points:

$$x_{min} = \min_{0 \leq i < n} x_i \quad (3.9)$$

$$y_{min} = \min_{0 \leq i < n} y_i \quad (3.10)$$

$$x_{max} = \max_{0 \leq i < n} x_i \quad (3.11)$$

$$y_{max} = \max_{0 \leq i < n} y_i \quad (3.12)$$

The area of the bounding box is then given by:

$$f_{BoundingBoxArea} = (x_{max} - x_{min}) \cdot (y_{max} - y_{min}) \quad (3.13)$$

This thesis combines different families of features, to exhaustively cover all the aspects of sketch characteristics encountered in the literature. A further categorization of features reflects their sensitivity to different properties of the stroke patterns, which can ultimately lead to more transparent and explainable ML models. Two major categories of features are introduced: *syntactic* features and *semantic* features. We distinguish each feature to be either a syntactic or semantic feature. **Syntactic features** reflect task-independent characteristics about the geometry and other aspects of the raw pen input and are divided into seven sub-categories (see table 3.1): angle-based, space-based, centroid-based, temporal, pressure-based, trajectory and meta features. These types of features can be applied to all sketch samples, independent of their source. In contrast, **semantic features** describe closely task-related knowledge, which is highly dependent on the use case. Semantic features are not in the focus of this thesis, as they need to be modeled for each application individually and therefore cannot be generalized easily across different data sets. For comparison sections 3.2.3 and 3.3 provide explanations and several examples of semantic features used in the analysis of the CDT.

Geometric features	
Angle-based	Space-based
Circular variance	Stroke length (avg./SD)
Rectangularity	Gesture length
Curvature (avg./SD)	Perimeter length

Angles after resampling Cosine/sine of first to last point vector Cosine/sine initial vector Bounding box diagonal angle (Signed) Perpendicularity (avg./SD) k-Perpendicularity Absolute/squared curvature Maximum k-angle Absolute directional angle Relative angle histogram Principal axis orientation (cosine/sine) Maximum angular difference Circular variance Sum of (absolute/squared) angles Macro perpendicularity (avg./SD)	Compactness Eccentricity Principal axes First/Last point X & Y Distance of first to last point First to last point vector Initial/final horizontal/vertical offset Average length of straight lines 2D histogram Ratio of axes Ratio of principal axes Length of first principal axis Convex hull area/compactness Sample ratio octants
<i>Centroid-based</i>	
Deviation Centroid offset & radius (avg./SD) Hu moments	
Temporal features	
Maximum speed (squared) Duration of gesture Pen-up/pen-down ratio Velocity (max/avg./SD) Acceleration/deceleration (max/avg./SD)	
Pressure-based features	
Pressure (max/avg./SD)	
Trajectory features	
Closure Inflexion X & Y Proportion of downstroke trajectory Ratio between half-perimeter and trajectory Stroke direction (avg./SD) Cup count & offset (first/last) Cosine/sine chain code	
Meta features	
Number of strokes & straight lines (SD) Straight-line ratio Largest straight-line ratio Number of connected components & crossings	

Table 3.1: Categorization of syntactic features, avg. = average, SD = standard deviation.

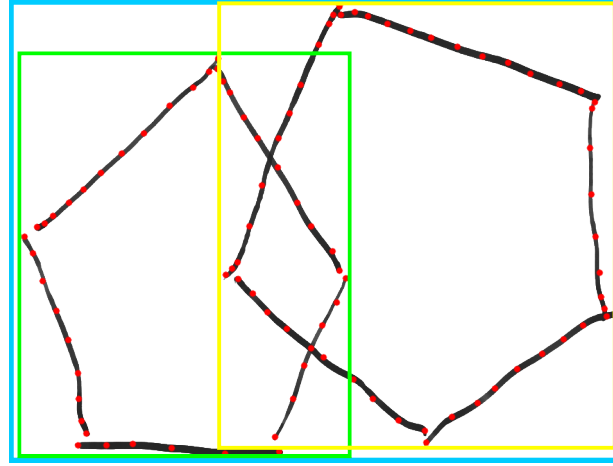


Figure 3.1: Visualization of rectangular bounding boxes around strokes (black) in the context of the "draw two overlapping pentagons" task during the MMSE cognitive assessment. The overlapping rectangular bounding boxes around the two individual gestures (green & yellow) and the larger bounding box (cyan). Red dots represent the sampling points recorded by the digital pen.

3.2.2 Syntactic Features

This work introduces the following seven categories of syntactic features:

(1) Angle-based features

Angle-based features are calculated from angles between sample points (e.g., curvature, perpendicularity, rectangularity).

$$\theta_i = \arccos \left\{ \frac{\overrightarrow{s_{i-1}s_i} \cdot \overrightarrow{s_i s_{i+1}}}{\|\overrightarrow{s_{i-1}s_i}\| \cdot \|\overrightarrow{s_i s_{i+1}}\|} \right\} \quad (3.14)$$

$$f_{Curvature} = \sum_{i=1}^{n-2} \theta_i \quad (3.15)$$

$$f_{Perpendicularity} = \sum_{i=1}^{n-2} \sin^2(\theta_i) \quad (3.16)$$

In ML-based gesture recognizers, these features can be used to distinguish rectangles from circles and ellipses, for example.

(2) Space-based features

Space-based features depend on the distances between samples (e.g., convex hull area, principal axes, compactness). The area A of a gesture is usually derived from the area

of the convex hull around all sample points, which can be calculated using Graham's algorithm [38]:

$$f_{ConvexHullArea} = A \quad (3.17)$$

With the area of the convex hull and the length of its perimeter l a feature called compactness is calculated. The closer the sample points are together, the smaller the compactness will be. Handwritten texts, e.g., will have a larger compactness than geometric symbols, such as rectangles [108]:

$$f_{Compactness} = \frac{l^2}{A} \quad (3.18)$$

As a result, this feature is used in pen-based intelligent user interfaces to distinguish the sketched content (e.g., gestures and symbols) from the written content (e.g., words and numbers).

The side length of the bounding box of a figure is used to calculate the eccentricity. Note that the co-ordinate axes are used instead of the principal axes (which are rotated with the pen gesture).

$$f_{Eccentricity} = \sqrt{1 - \frac{b^2}{a^2}} \quad (3.19)$$

In handwriting and gesture recognition this feature is commonly used to distinguish a circle from an ellipse.

(3) Centroid-based features

Centroidal features describe relations between sample points and the overall centroid (e.g., centroid offset, deviation, average radius). Using the dimensions of the bounding box the center point c can be calculated:

$$c = \begin{pmatrix} x_{center} \\ y_{center} \end{pmatrix} = \begin{pmatrix} x_{min} + 0.5 \cdot (x_{max} - x_{min}) \\ y_{min} + 0.5 \cdot (y_{max} - y_{min}) \end{pmatrix} \quad (3.20)$$

The average distance of sample points to the center point is another feature:

$$f_{MeanCentroidDistance} = \frac{1}{n} \sum_{i=0}^{n-1} \|s_i - \mathbf{c}\| \quad (3.21)$$

In the case of the CDT, for example, one of the evaluation criteria is the extent to which all numbers are evenly distributed, which can be calculated with the distance from the center.

(4) Temporal features

Temporal features are derived from timestamps of sample points (e.g., duration, speed, acceleration). The velocity v between sample points is defined as:

$$v_i = \frac{s_{i+1} - s_{i-1}}{t_{i+1} - t_{i-1}} \quad (3.22)$$

From which the feature of average velocity is calculated:

$$f_{AverageVelocity} = \frac{1}{n-2} \sum_{i=1}^{n-2} \|v_i\| \quad (3.23)$$

The acceleration is calculated as follows:

$$a_i = \frac{v_{i+1} - v_{i-1}}{t_{i+1} - t_{i-1}} \quad (3.24)$$

And the average acceleration is then given by:

$$f_{AverageAcceleration} = \frac{1}{n-4} \sum_{i=2}^{n-3} \|a_i\| \quad (3.25)$$

These characteristics allow for an analysis of sketch areas in which the subject took more time, e.g., areas in which he was more concentrated or found the task to be more difficult. Changes in writing style can also be measured over time.

(5) Pressure-based features

Pressure-based features are computed from hardware sensors capturing applied pressure (e.g., average pressure, standard deviation). The most intuitive features are the average pressure and the standard deviation in pressure:

$$f_{AveragePressure} = \frac{1}{n} \sum_{i=0}^{n-1} p_i \quad (3.26)$$

$$f_{StandardPressureDeviation} = \sqrt{\frac{1}{n} \sum_{i=0}^{n-1} (p_i - f_{AveragePressure})^2} \quad (3.27)$$

Since temporal and pressure-based features are difficult to detect with the human eye, a further component for the visualization of features is developed as part of this thesis. Figure 3.2 shows the result of the visualization for pressure-based based features of two figures from the CERAD neuropsychological test battery. The color coding of the feature values at each data point allows the physician to analyze additional properties, which would not be measured during a traditional assessment and can provide additional information about the cognitive state of the patient.

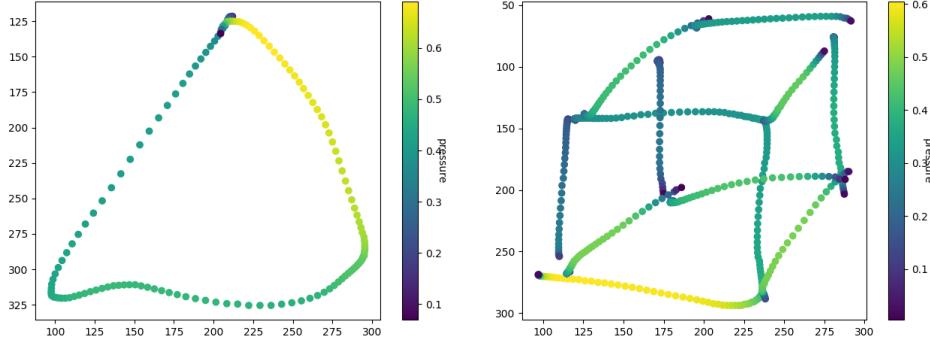


Figure 3.2: Visualization of pressure-based features. The digital pen, which was used to record these samples, provides a floating point pressure value in the range of 0 to 1.

(6) Trajectory-based features

Trajectory-based features reflect the visual appearance of strokes (e.g., closure, average stroke direction).

The path length from one sample point to another is denoted as L and is calculated as follows:

$$L_{i,j} = \sum_{k=i}^{j-1} \begin{cases} 0 \\ \|s_k s_{k+1}\| \end{cases} \quad (3.28)$$

$L = L_{0,n-1}$ is the total length of S . Whereas the first to last point vector and its length is:

$$v = \overrightarrow{s_1 s_n}, \quad \|v\| = \|s_1 s_n\| \quad (3.29)$$

Typical trajectory-based features are closure and average direction:

$$f_{closure} = \frac{\|v\|}{L} \quad (3.30)$$

$$f_{AverageDirection} = \frac{1}{n-1} \sum_{i=0}^{n-2} \arctan \left(\frac{y_{i+1} - y_i}{x_{i+1} - x_i} \right) \quad (3.31)$$

In the case of the TMT, for example, these features can be used to analyze the direction of the strokes and thus the order of the connected nodes.

(7) Meta features

Meta features are higher-level features and relations between components (e.g., number of strokes, inter-connections, crossings, straight-line ratio). One intuitive example is the number of straight lines ($f_{\#StraightLines}$) or, to be more precise, the number of straight segments. A sliding window with a threshold is used to calculate sets of connected points which have minimal curvature between them. The size of the sliding window and threshold can be either dynamically adjusted to the length of the stroke or be a fixed value depending on the task.

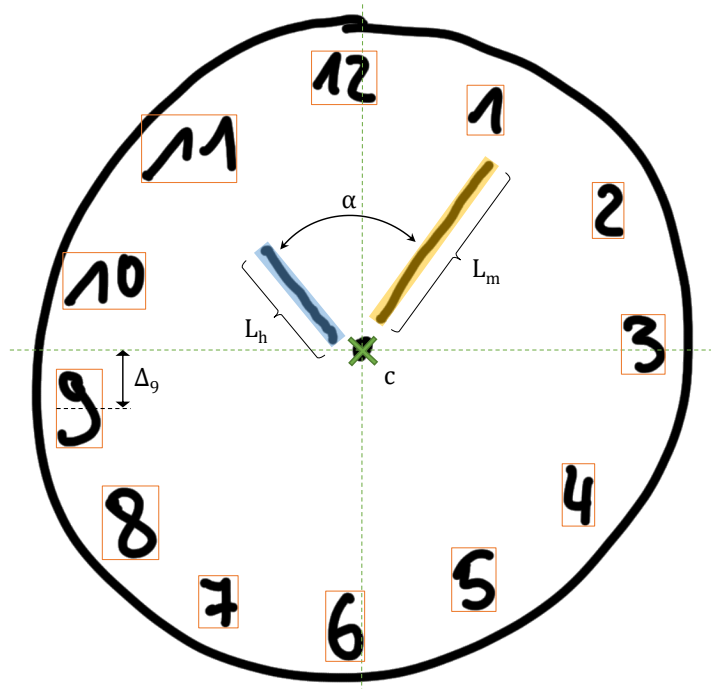


Figure 3.3: Visualization of semantic features in the context of the CDT.

3.2.3 Semantic Features

Depending on the task, additional features can be constructed from the task description. As these features describe higher-level semantic concepts about the sketched contents, they are referred to as *semantic features*. Semantic features highly depend on the given context and therefore vary noticeably between different tasks. Such features usually cannot be transferred easily to other use-cases, as they are often hard-coded for each application. The next section (3.3) describes such a system.

Figure 3.3 shows the visualization of a selected semantic feature set in the context of the CDT. Participants are asked to draw a clock face with the time set to 10 past 11 o'clock. The drawn clock is then examined by a trained physician and rated based on a predefined scoring scheme, reflecting the visual appearance and integrity of the clock using a numerical score. Some of the features found in scoring systems are quantitative, such as checking for the presence of a digit or a hand. Others are less well defined: for example, one scoring item calls for determining whether the minute hand is "obviously longer" than the hour hand, while another checks whether there are "slight errors in the placement of the hands". These can be estimated by a physician, but it is not immediately obvious how to compute them as mathematical features in a way that captures the original intent. Without providing quantitative definitions of those terms, the results may vary in scoring and analysis depending on the experience of the physician. Since most of the scoring systems have been designed to be quick and easy to be interpreted by human testers, they are not all equally well suited for automation. Here, the detailed 20-point Clock Drawing Interpretation Scale (CDIS) by Mendez et al. [60] is selected, which is well suited for automation because it contains clear test parameters that can be modeled mathematically and computationally. In addition, the manual scoring procedure of the

CDIS is very time-consuming and would highly benefit from automatic computation. In this example, the following semantic features are calculated based on the CDIS scoring system (see figure 3.3):

- c denotes the center point of the clock (centroid), the closer it is to the center of the clock's circle, the more points are awarded.
- L_h and L_m represent the lengths of the hour and minute hands respectively. If the clock is well drawn, the hour hand should be shorter than the minute hand.
- The angle between the hour and minute hands is denoted as α . Together with the orientation of the hands, it can be used to determine if the correct time was set.
- Δ_9 is the displacement of clock face digits relative to their ideal location. In this example it is the vertical offset of digit number 9 to its correct center position.

In the context of the TMT, the semantic features could describe whether nodes were connected, or whether or not their connection order was correct. Similarly, the appearance of the 18 sub-figures of the ROCF can be modeled (e.g., their positioning and visual appearance) as required by the scoring scheme. Several of the automated scoring systems described in section 2.4 (*Analyzing Cognitive Tests using Digital Pen Features*) are based solely on semantic features or benefit from including them into their analysis. Semantic features are also applicable in other domains, such as education, where sketches of diagrams and mathematical formulas convey particular semantic meanings, or pen-based interactive systems, where compound gestures (e.g., annotations, correction gestures) are used to trigger specific actions. However, since the majority of semantic features is not generalizable between different use cases, the main focus of this thesis lies instead on the syntactic features, which can be applied independent of the task at hand, and do not need to be re-implemented for each type of application.

3.3 Automated Scoring of the CDT

This section describes a system for the automated analysis of the CDT that utilizes a combination of syntactic and semantic digital pen features. The described system is part of the Interakt project and was published in the proceedings of the ACM UMAP conference on User Modeling, Adaptation and Personalization in 2019 [79]. It implements the Mendez scoring scheme and creates a hierarchy of error categories that model the test characteristics of the CDT, based on a set of impaired clock examples provided by a geriatrics clinic. Using a digital pen, a total of 120 clock samples are recorded for the evaluation of the automatic scoring system. Results show that the system provides a clinically relevant cognitive model for each subject. In addition, the time spent on manual scoring is heavily reduced.

The technical architecture of the system is shown in figure 3.4. Clock sketches are recorded using a digital pen and paper imprinted with a nearly invisible microdot pattern. The NeoSmartpen N2⁵ is a ballpoint pen with an integrated infrared camera near the tip, which recognizes the microdot pattern on the paper and records the exact position, timestamp and pressure of the pen. It streams the raw stroke data via Bluetooth to a connected tablet or smartphone. In the backend, each stroke is then analyzed and annotated with a corresponding gesture type (such as circle, hand, number etc.).

⁵<https://www.neosmartpen.com/>

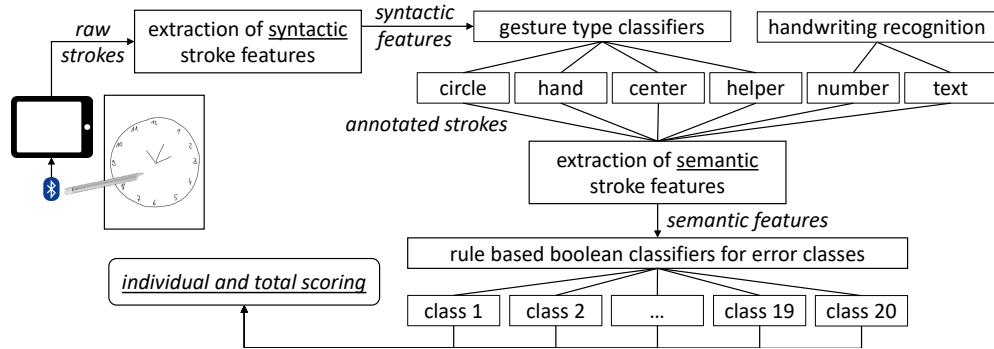


Figure 3.4: Architecture of the classification system for the automatic scoring of the CDT.

This is done by calculating the 165 syntactic features on each of the strokes and using a rule-based gesture type classifier in combination with an off-the-shelf handwriting recognition engine to annotate each stroke. Next, the values of the 20 CDIS scoring classes are computed according to Mendez et al. [60]. To compute the scoring classes, 20 rule-based classifiers are created, which correspond to the conditions of the CDIS scoring items. The rule-based classification relies on a combination of syntactic and semantic features. A subset of stroke-level syntactic features is used to cover geometrical properties of the sketched strokes, such as length, curvature or compactness (see section 3.2.2). In addition, a set of semantic features is extracted as described in the previous section. For example, the distance to the clock center, or the location of a number inside a clock quadrant are used as semantic features to predict the CDIS classes. The individual scores produced by the classifiers are then summarized to create the final CDIS score, which indicates the overall level of cognitive impairment indicated by the drawing.

The system differentiates between seven gesture types. A stroke can belong to a part of the circle, digit, hand, center, helping line, text and unknown/uncategorized. In an iterative streaming process, the system looks at stroke patterns that match one or more of the gesture classes based on the stroke-level features. For example, a straight line has the same initial and final directional vector as a circle, whereas parts of the circle show a distinct curvature. Other gestures are classified based on their spatial location, e.g., markings on the outer circle are classified as being helper markings. For the detection of digits and text, a commercial handwriting recognition engine by MyScript⁶ is used. An intuitive example for the classification process is CDIS item 10 (“All numbers 1-12 are indicated.”), where the handwriting recognition engine is used to label the strokes with the corresponding numbers. Then the system iterates through all strokes classified as numbers and checks if all are present. Other scoring items require the use of additional features, such as item 18 (“There are exactly two distinct and separable hands.”). The strokes must first be labeled as hands, then the algorithm compares their length to predict the scoring result. Some scoring items, e.g., item 14 (“All symbols lie about equally adjacent to a closure figure edge.”) still leave room for interpretation. These classes are implemented using a mixture of thresholds and mean values.

⁶<https://www.myscript.com/>

Chapter 4

Data Collection Method

The following chapter details how the sketch data sets, used for evaluation in chapter 5, were obtained, and what kind of sketches they contain. The *Handwriting & Sketch Recognition Data Sets* section describes the data sets to be used for ML-based benchmark experiments (i.e., the evaluation and comparison of different feature sets in handwriting and sketch recognition). In contrast, section *Cognitive Assessments Data Set* elaborates on how the data for ML-based cognitive performance classification is recorded using a pen-based interactive cognitive assessment tool.

4.1 Handwriting & Sketch Recognition Data Sets

Table 4.1 provides an overview of all ten sketch data sets considered in this thesis. The selection covers a wide variety of symbols, gestures and sketches from different domains and pen input devices. The first eight data sets are selected because they are part of the HBF49 baseline experiment [25]. The *DFKI-symbols* set is recorded as part of this thesis and *How Humans Sketch* is a recent set of sketches with a large number of classes and participants, therefore ideally suited to complement this selection. Several of these sets are freely available for download, while others need to be paid for. Obtaining a copy of the *Niclcon* data set was particularly difficult because sadly the first author passed away a few years ago and his assistant had not retained a copy. Fortunately, the authors of the HBF49 benchmark were kind enough to share their data.

Another issue is that out of all the data sets only two share the same file format. In addition, only two of the data formats are officially standardized (JSON & SVG), therefore individual parsers need to be implemented to convert the data samples into the format of the digital pen features library. Furthermore, not all meta information (number of samples, subject, etc.) about the data sets is the same between the original publications, the specifications of the HBF49 benchmark experiments and the available data in the downloaded data sets. For example, the LaViola data set has 11,602 samples according to the HBF49 publication [25], but the official download only contains 10,655 samples. This is why, table 4.1 contains numbers based on the downloaded data sets, which may differ from the benchmark and original publication.

Data set	Classes	Subjects	Samples	Timestamps	Pressure
CVCsymb [87]	25	12	4,278	✗	✗
HHReco [46]	13	19	7,410	✓	✗
ILG [86]	21	27	4,704	✗	✓
IMISketch [50]	13	-	1,871	✗	✗
Ironoff-digits [105]	10	411	4,086	✓	✓
LaViola [56]	51	11	10,655	✗	✗
NicIcon [68]	14	34	24,441	✓	✓
Sign [4]	17	21	33,154	✗	✓
DFKI-symbols [76]	11	9	9,938	✓	✓
How Humans Sketch [29]	250	1,350	19,995	✗	✗

Table 4.1: Comparison of data sets for handwriting and sketch recognition ML experiments (✓ = data available, ✗ = no data available).

4.1.1 CVCsymb Data Set

The first data set considered for the experiments is the multi-stroke CVCsymb set published by Romeu et al. [87]. It contains symbols used in architectural and electrical blueprints, such as tables, beds, doors, sinks and TVs (see figure 4.1). The set was recorded using a digital pen as part of a rule-based sketch recognition tool. It is composed of 25 symbols drawn by 12 people, with a total of 4,278 samples (171 samples/class). Compared to other data sets, the recordings have a high sampling rate with less noise.

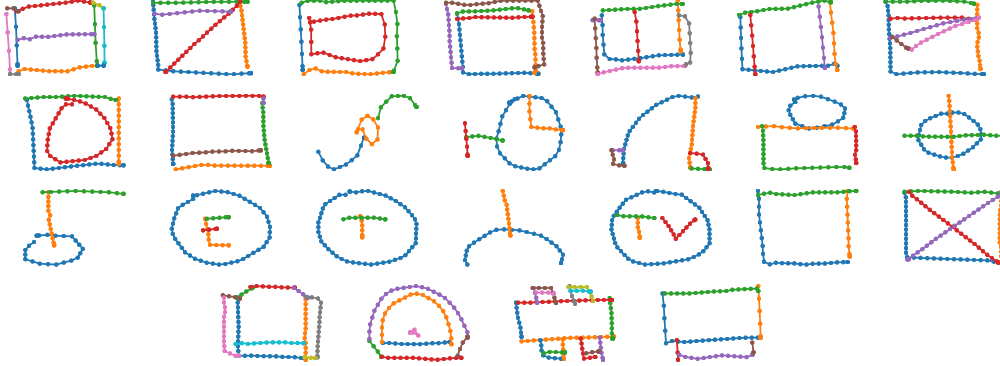


Figure 4.1: Symbols of architectural and electrical blueprints from the CVCsymb data set.

4.1.2 HHReco Data Set

Published by Hse & Newton [46], the HHReco data set contains symbols commonly used in office tools (e.g. UML diagram editors, slide drawing programs like Microsoft PowerPoint, or electrical schematic editing tools). The set of shapes was selected based upon the applications of interest, commonly used basic shapes, and the geometric properties of shapes (e.g., shapes with lines, shapes with curves, shapes with mixed lines and curves, and shapes with and without self-intersections). It contains samples from 19 users. Each user was asked to sketch 30 examples for each of the 13 symbols shown in

and number of strokes). In total, 13 types of symbols are included (144 samples/class), ten of which are furniture symbols and three are types of openings.

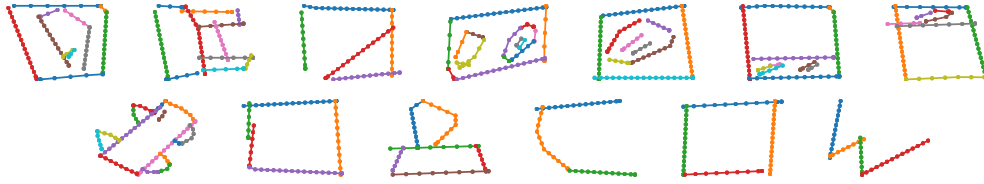


Figure 4.4: Architectural symbols of the IMISketch data set, vectorized from image data.

4.1.5 Ironoff-digits Data Set

Viard-Gaudin et al. [105] have collected data of isolated characters, digits and cursive words as part of the Ironoff data set. They used a Wacom UltraPadA4 graphic tablet to record input samples of A4-forms with predefined text boxes. Above each box, the ground truth of the character or word to be written was provided. Each form has been filled by a different writer, totaling in 4,086 samples from 411 participants (409 samples/class). Following the HBF49 benchmark experiment [25] only the data of forms containing 0-9 digits is considered, resulting in ten classes (see figure 4.5).

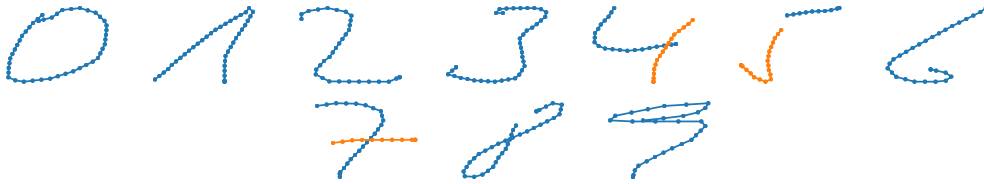


Figure 4.5: Samples of the 0-9 digits from the Ironoff-digits multi-stroke data set.

4.1.6 LaViola Data Set

LaViola et al. [56] collected handwriting samples from 11 subjects, recorded with a Hewlett-Packard Laboratories (HP) Compaq tc1100 Tablet PC. The set contains 48 different symbols including *a-z*, *0-9*, Σ , $(,)$, $-$, $\sqrt{}$, \int , $\{$, $<$, $>$, $+$, \neq and *else*. Figure 4.6 depicts a subset of these multi-stroke symbols. LaViola et al. chose to include mathematical symbols as part of their set because it was designed to be part of a mathematical expression recognizer. In total their data set consists of 10,655 symbols (209 samples/class).



Figure 4.6: Subset of multi-stroke sketch samples from the LaViola data set.

4.1.7 NicIcon Data Set

NicIcon by Niels et al. [68], is a set of 14 icons from the domain of crisis management and incident response. The icons were designed such that they have a visual resemblance to the objects they represent or correspond to well-known symbols (see figure 4.7). In total, 32 people participated in the experiment, each of whom was asked to fill out 22 paper forms, using an inking stylus on paper. The paper forms contained seven rows of five columns, resulting in 35 drawing areas. Each of the 32 participants had to fill in 22 paper sheets, resulting in 770 symbols per participant. Some participants skipped certain gestures, so the total number of samples is 24,441 [68] with approx. 1,746 samples per class.

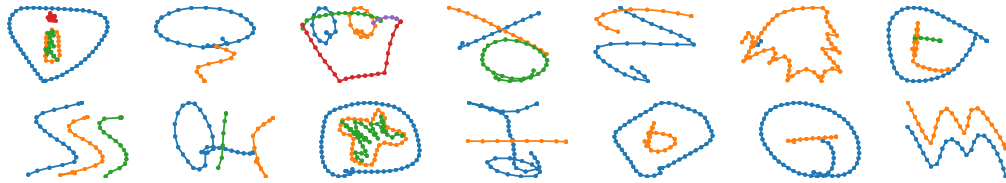


Figure 4.7: The 14 NicIcon symbols from the domain of disaster response.

4.1.8 Sign Data Set

The Sign data set [4] contains 17 different handwritten gestures (see figure 4.8), drawn by 20 participants on Tablet PCs. During data acquisition each participant performed four sessions, where each session starts with drawing each gesture five times, and then gestures to draw are presented in a random order to the participant, to simulate the use of an application. Each gesture is drawn 100 times by each participant, resulting in about 1,700 gestures per participant and a total of 33,154 gestures [4] with approx. 1,950 samples per class.

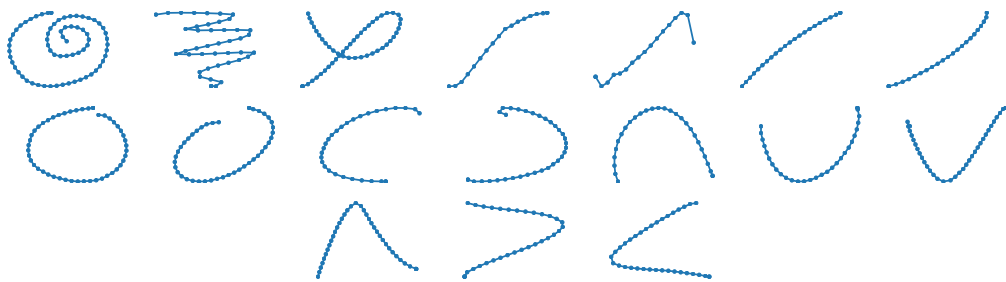


Figure 4.8: Gestures for commands in software applications as part of the Sign data set.

4.1.9 DFKI-symbols Data Set

Whereas the previously described data sets were chosen as part of the HBF49 benchmark experiments, the next set is recorded as part of the cognitive testing use case. Based on the design of sketching tasks in neurocognitive testing batteries, a set of 11 gestures is created. These are gestures, which are commonly found in different cognitive assessments [76].

The focus is on the geometric shapes of which cognitive tests are composed, e.g., the CDT contains a circle (clock face) and lines (hands). The CERAD battery, MMSE and MoCA contain several shapes like pentagons, diamonds and rectangles. Single shapes, in turn, compose parts of other assessments, such as the ROCF, which contains several sub-shapes, including, triangles, rectangles, lines and circles. Figure 4.9 shows the 11 classes of the DFKI-symbols data set. A total of eight shapes were chosen from the most commonly used cognitive assessments: arrow, circle, rectangle, triangle, circle, diamond, overlapping rectangles, cube and overlapping pentagons. Three additional gestures are chosen based on a previously conducted user study, where participants were asked to specify gestures that they would use to indicate completion of the current handwriting task. Such symbols can be used in pen-based interfaces for assisted living, e.g., SMS or e-mail writing applications for the elderly, to indicate that the message should be sent [78]. Our symbol data set consists of 11 classes (shapes) with 100 samples per class, recorded on an A4-sized Samsung Galaxy Note tablet with a stylus. Overall, nine subjects have provided a total of 9,900 handwritten samples with 900 samples per class.

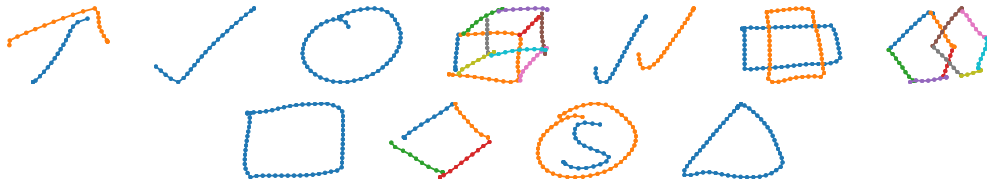


Figure 4.9: The 11 symbols of the DFKI-symbols data set.

4.1.10 How Humans Sketch Data Set

The last data set to be considered here was recorded by Eitz et al. [29] as part of a large-scale exploration of human sketches. It contains a set of 250 categories based on everyday objects, such as airplanes, alarm clocks, mugs, radios, animals and many more. Figure 4.10 presents a selection of 21 sketches from these categories. The sketches were recorded using crowdsourcing, with a total of 1,350 participants. This data set is relevant, for this thesis, because it contains a large number of classes & participants, and each of the sketches is composed of many strokes (approx. 13 strokes per sketch). A drawback is the lack of temporal and pressure information from the input signal and the relatively low fraction of 80 samples per class.

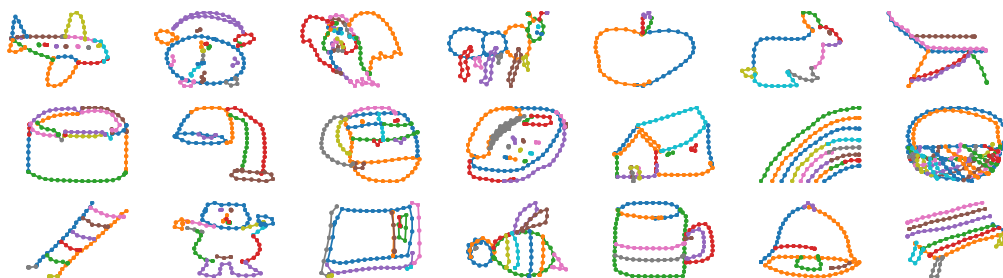


Figure 4.10: A selection of 21 sketch classes from the How Humans Sketch data set.

4.2 Cognitive Assessments Data Set

In the medical domain, pen-based neurocognitive testing is used to assess a wide range of cognitive impairment, including dementia, Parkinson's disease or traumatic brain injury. Usually a trained professional observes the test and scores the final sketch based on a scoring scheme, which takes up to a few minutes. An automatic scoring system has several benefits. First, it significantly reduces the time caregivers have to invest in administering the assessment. Second, it is likely to produce more objective scores and potentially enables a more detailed analysis. Therefore, this section presents a cognitive assessment tool that is used to record a sketch data set of cognitive assessments with the goal to automatically analyze test performance based on digital pen features.

4.2.1 Interactive Cognitive Assessment Tool Architecture

Figure 4.11 provides a general overview of the automatic cognitive assessment tool for the CDT, TMT and ROCF. It is fully implemented and currently being deployed in a geriatrics daycare clinic in Berlin. As prerequisite, digital versions of the original assessment forms are created by scanning and then overlaying the original with reference points and bounding boxes, e.g., for each of the encircled nodes of the TMT. The Neo smartpen N2⁷ and its Euclidean 2D coordinate space are used to record the sketch samples. With a built-in infrared camera, the N2 digital pen tracks its position on the piece of paper, by recognizing a micro-dot pattern, that is printed on the paper forms. The pattern is almost invisible to the human eye and can be printed by standard laser printers, such as they are present in hospitals, clinics and medical offices. This approach does not require special software, the forms are printed directly from a PDF file. A physician connects the digital pen via Bluetooth to a mobile application, which can be run off any common Android smartphone or tablet and is connected to the hospital information system. Recorded handwriting data is streamed in real-time to the tablet, where the digital pen strokes are directly rendered on the screen, giving instant feedback to the physician. This setup has the potential benefit that physician and subject do not need to sit close together, e.g., in telemedicine applications or due to Covid-19 distancing regulations.

The raw pen data is exported to a backend service, which creates a digital copy of the document for indexing and documentation purposes. The pen data processing server analyzes the pen strokes as a series of time-stamped two-dimensional coordinates. In case of the TMT, it matches the input to the reference template of the original TMT, thereby reproducing the path taken by the subject (see figure 4.13b). During this process, the algorithm produces transparent explanations for encountered errors and extracts missing connections. The completion time, which is the major scoring criterion for the TMT, is calculated for each sample by subtracting the timestamp of the first recorded pen stroke point from the last recorded timestamp. For the CDT the strokes are segmented into semantic classes, such as digits, helper lines, center point, hands etc. (see figure 4.12b). Based on this segmentation the system generates scores following the 20-point CDIS scoring scheme by Mendez [60]. Similarly, the ROCF sketches are segmented into the 18 sub-figures of the ROCF (see figure 4.14b) and scored according to the official scoring scheme, whose score is based on the individual scores of the sub-figures [15, 28].

For each subject, the results of the scorings are summarized in a structured PDF document, which includes the completion time, the original rendering of the recorded data, the annotated version of the analyzed data and detailed information about the scoring

⁷<https://www.neosmartpen.com>

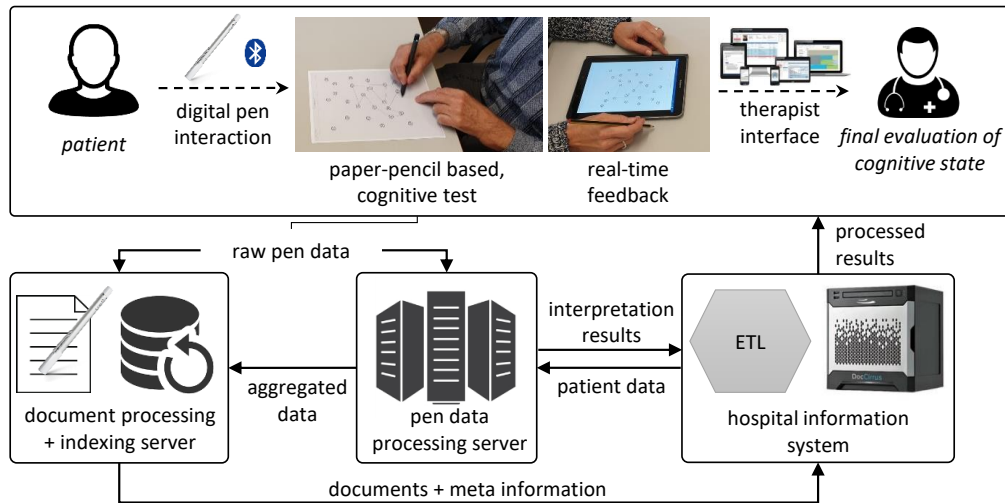


Figure 4.11: Architecture of the interactive cognitive assessment tool. Data is recorded using a digital pen and streamed to a backend service, where it is analyzed before being loaded into the hospital information system and presented to the physician. ETL = extract, transform, load; a general concept of automatic data processing.

results. Providing a rendering of the unaltered sample serves both documentation and transparency requirements: physicians need to be able to examine the original and to have a clear understanding of what was generated by the system. To further increase the transparency of the automatically generated scoring, the physician is presented with a representation of what the algorithm detected, e.g., for the TMT the reference points are indicated in green and strokes are colored in red when recognized as not connecting any two nodes (see figure 4.13b). For the CDT and ROCF, the segmented sketches are included (see figure 4.12b and figure 4.14b). This kind of transparency is highly relevant to the physicians, as they need to be able to judge whether the automatic scoring, they are relying on, is accurate and can be trusted. Similarly, the focus of this thesis is on the analysis of digital pen features to predict test performance using ML, which can be used to produce models that can be explained by the features themselves.

4.2.2 CDT, TMT & ROCF Data Collection Method

The data set is collected during a repeated measures experiment with 40 participants, who are recruited from the geriatrics daycare clinic of a large German hospital in Berlin. Inclusion criteria for participants are a minimum age of 65 and the signed informed consent. Exclusion criteria are severe cognitive disorders, mental diseases, severe auditive, visual, linguistic, sensory or motor limitations, chronic pain or a legal representative. A total of 40 older adults are included (age $M=74.4 \pm 4.1$ years, range: 67–85 years) in the experiment, of which half are female. The majority of participants is well-educated (57.5% high-level education) and right-handed (95.0%). A total of 25 participants (62.5%) rate their health as rather or very good, whereas 22 participants (55.0%) report to suffer from a chronic disease such as diabetes ($N=8$) or hypertension ($N=5$). When asked afterwards, whether the type of pen influences their test performance, almost all participants (95.0%) answer the question in the negative.

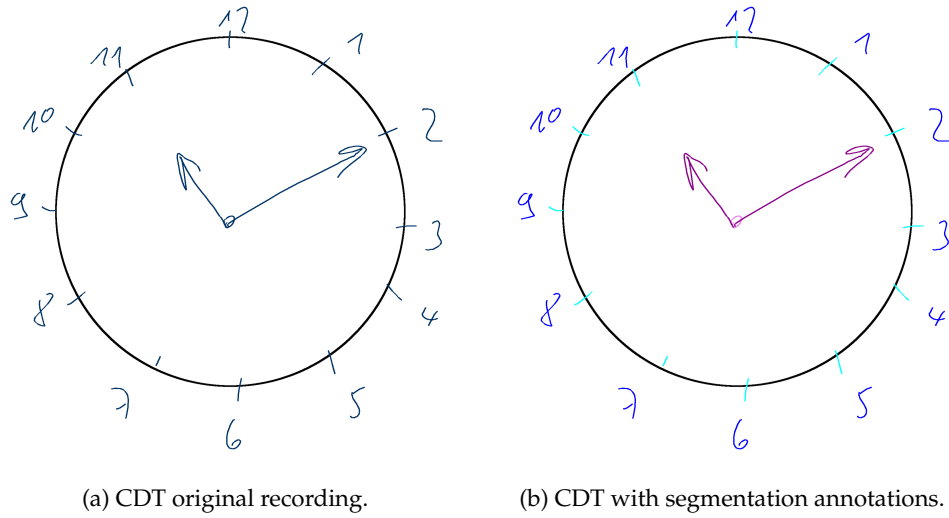


Figure 4.12: Renderings of the original input (a) as recorded by the digital pen during the data collection with a subject performing the CDT. The automated system analyzes the input and segments the strokes into classes (b) (digits, helper lines, center point, hands etc.). Using this segmentation the system automatically performs a scoring based on the 20-point CDT scoring scheme by Mendez [60].

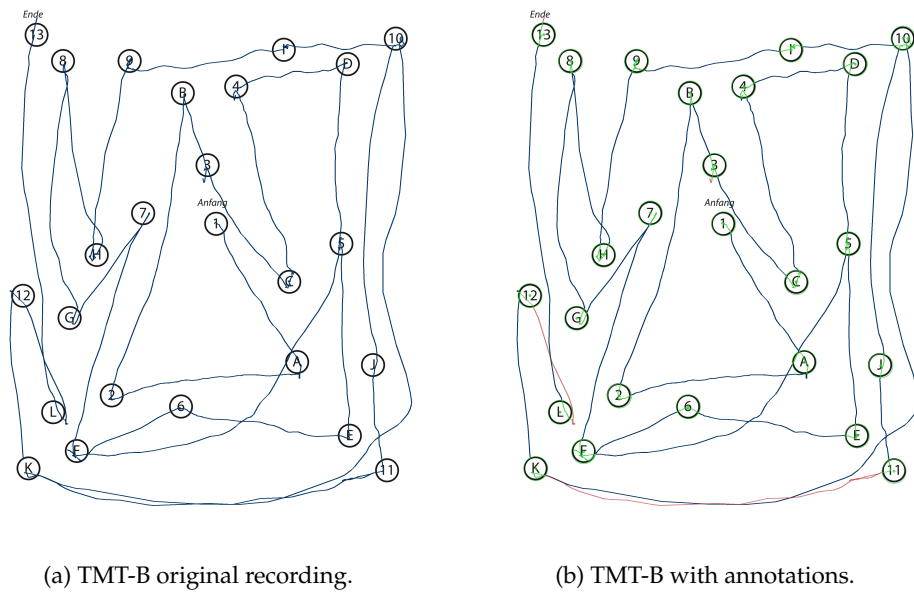


Figure 4.13: Renderings of the original input (a) as recorded by the digital pen during the data collection with a subject performing the TMT-B. The automated system analyzes the input and annotates the stroke data with the information relevant for the algorithm's final scoring decision (b). Green dots indicate the center points of the circles and green stroke parts indicate that a circle has been successfully hit by a stroke, while red strokes could not be identified as connecting two circles.

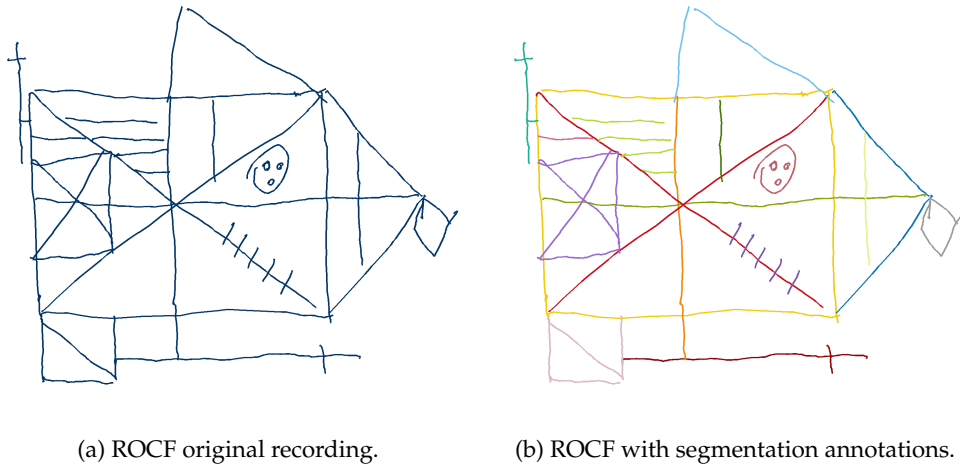


Figure 4.14: Renderings of the original input (a) as recorded by the digital pen during the data collection with a subject performing the ROCF. The automated system segments the strokes into the 18 ROCF sub-figures, which are represented by different colors (b). Based on this segmentation the algorithm scores the individual sub-figures, from 0 = "not present", to 2 = "correct" [28].

The experiment has two conditions, paper-pencil and digital pen on paper. The versions of the cognitive assessments are the same in both conditions, just the type of pen differs. For each condition, participants perform a total of 6 tasks (CDT, TMT (A and B), and ROCF (copy, immediate recall and delayed recall)). To avoid order effects, the execution order is balanced: half of the participants start with the paper-pencil version first and the other half with the digital pen. Participants sit in a distraction-free room at a table with the pen and the test in front of them (see figure 4.15). The physician sits on the opposite side of the table and holds a tablet, to which the digital pen is connected and on which the participant's input can be tracked in real-time. The feedback is only visible to the experimenter. The tablet records the data, which is streamed via Bluetooth by the digital pen, for later analysis. An important factor for successful deployment is the seamless integration of digital medical applications into everyday hospital processes. The system supports this requirement in several ways. First, the tablet application and digital pen are highly mobile and can directly be taken into the examination room, or to the patient's bedside, if necessary. Second, the application has access to a list of patients from which the physician can choose and initiate an assessment, thereby directly linking the digital test to the correct patient file. If required for the analysis, the backend service can access patient data, such as demographic data (e.g., age) or previous assessments for a comparison of cognitive performance at different points in time. Third, all structured reports, generated during analysis, are sent to the hospital information system, these include the generated PDF report, the same information in a structured file format for direct use in database systems and a replay video of the digital pen recording. Lastly, the physician can access and query all information in a web-based user interface.

Participants are given the original test instructions for each task [28, 34, 84]. For each condition they are asked to first perform the ROCF (copy and immediate recall), then the TMT-A and TMT-B (the order is given by the TMT's test design), followed by the CDT and finishing with the ROCF (delayed recall). By letting participants perform the other tests in between the ROCF tasks, the physician makes optimal use of the time, as

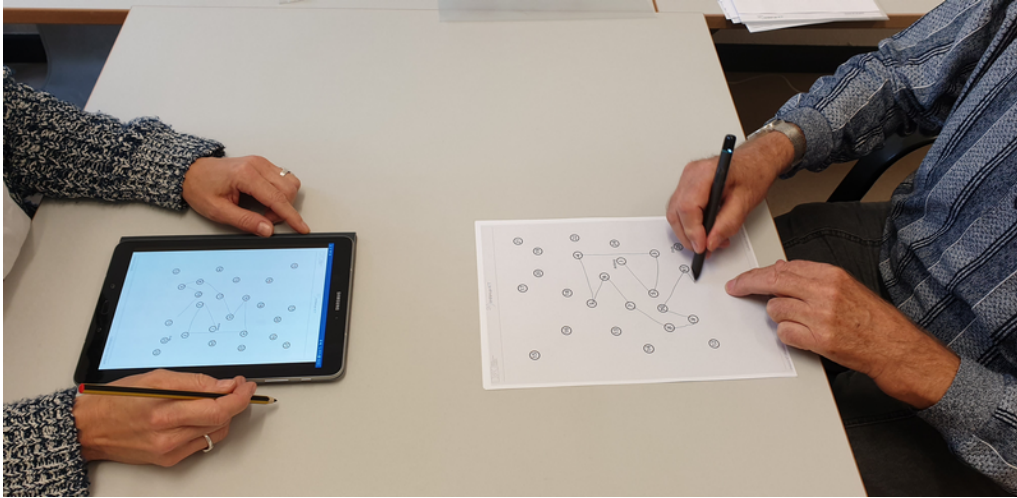


Figure 4.15: The data collection setup in the geriatrics daycare clinic. The subject (right) performs the cognitive test (here TMT) using a digital pen, which is connected via Bluetooth. The result can be monitored in real-time on the tester's tablet (left).

the test design requires a pause between recall and delayed recall. The instructions for the tests are given by a trained physician. In both modes (normal and digital pen) the TMT task execution time is measured manually by the rater using a stopwatch. There is a wash-out phase between the modes of approximately 30 minutes. During this phase, participants complete a self-developed questionnaire to collect socio-demographic data and a questionnaire to record the technology commitment [66].

The collected data set consists of a total of 240 sketches (40x CDT, 80x TMT and 120x ROCF). Four samples of the ROCF are discarded, in which participants did not sketch anything and for which therefore it is not possible to calculate any features. After removing noise (e.g., notes and scribble) from the sketches, the stroke coordinates are translated to the origin of the Cartesian coordinate system ($\min(x) = 0, \min(y) = 0$). No scaling or pre-processing is applied to the sketches to avoid influencing participants' sketching characteristics. Each sketch is analyzed manually by a physician in accordance with the medical guidelines. For the TMT the score is based on the measured execution time in seconds [83], while the 20-point Mendez CDIS scheme [60] is applied to the CDT and the original 36-point scoring system for the ROCF [28]. In the considered use case of geriatrics, the scorings are used as an indicator to assess whether or not to schedule further cognitive testing procedures. Therefore, each sketch is classified into one of two classes, healthy and suspicious, based on the scoring results. Dependent on the execution time the TMT-A and TMT-B samples are classified as suspicious if they take over a minute or over two minutes respectively. Samples of the CDT are labeled as suspicious if the score is 18 or less, whereas ROCF samples are considered suspicious with scores of 30 or less. In total 152 samples are labeled as healthy and 84 as suspicious.

Chapter 5

Evaluation

In this chapter, a series of four ML experiments is conducted to evaluate the performance of digital pen features for sketch recognition and cognitive test performance classification. First, the proposed set of 165 features is compared to the HBF49 benchmark (49 features) on eight different sketch data sets and two ML methods. Second, the features are used to compare the sketch classification performance of ten ML methods on ten sketch data sets from various domains. The third experiment tests whether applying AutoML on these ten data sets can improve the classification results. Lastly, the features are used to classify cognitive test performance in the use case of neurocognitive assessments, without performing content analysis, only based on the digital pen features. For each of these experiments the methodology, results and discussions are provided individually.

5.1 HBF49 Benchmark Comparison

Digital pen features are most commonly used for the task of classifying a set of strokes as belonging to a certain class. This can be for example as part of a handwriting recognition system, where different letters and digits need to be distinguished, or in a multi-stroke sketch recognition system, e.g., to differentiate symbols and gestures, or to predict complex classes, e.g., whether a sketch represents an airplane or an alarm clock. As presented in the related work section, various sets of features, sketch data sets and ML experiments have been published for sketch recognition tasks. Unfortunately, only a small fraction of these experiments presents reproducible feature definitions and repeatable ML experiments. Delaye & Anquetil introduce the HBF49 (Heterogeneous Baseline Feature) set [25], which contains 49 features and more importantly a transparently designed set of ML experiments on eight publicly available sketch data sets. In order to compare the here proposed set of 165 features to the 49 features of HBF49, the benchmark experiments are repeated. The main hypothesis is that the accuracy of the ML classification increases if a higher number of digital pen features is provided as input.

5.1.1 Methodology

The HBF49 benchmark compares 1-Nearest Neighbor (NN) and Support Vector Machine (SVM) classifiers on the following eight data sets: CVCsymb, HHRco, ILG, IMISketch, Ironoff-digits, LaViola, NicIcon and Sign. A detailed summary of these data sets is provided in section 4.1. For each of the data samples, a feature vector of 49 HBF features is calculated and used as input to train the ML classifiers. The labels of the individual data sets are used as the prediction targets, e.g., for the Ironoff-digits data set the classes correspond to the ten digits. The results are then compared in terms of recognition accuracy. All experiments are conducted as described by the HBF49 benchmark, with one minor difference. The HBF49 publication uses the Weka [41] and LIBSVM [18] frameworks, which were very popular at that time, whereas here the more modern scikit-learn [75] library is used. Scikit-learn is a free software machine learning library for the Python programming language [75]. It features various classification algorithms, including SVM, random forests, gradient boosting, and many more. Scikit-learn is designed for high-performance computation and seamless integration with popular scientific libraries, such as NumPy and SciPy. In addition, the SVM in scikit-learn is implemented through a wrapper for LIBSVM, which makes it an ideal choice for conducting the HBF49 experiments.

In the context of ML, overfitting refers to a model that models the training data too well and thus does not generalize well for unseen data. To avoid it, it is common practice in supervised ML experiments to hold out part of the available data as a test set, on which the prediction performance is measured. Ideally, the train and test sets are distinct. The choice of how to split the data set into training and test data is vital for the performance of most supervised ML algorithms. However, by partitioning the available data into subsets, the number of samples which can be used for learning the model is drastically reduced, and the results can depend on a particular random choice for the pair of train and test sets. The HBF49 experiments solve this problem by employing a common procedure called cross-validation (CV). In the basic approach, called k -fold CV, the training set is split into k smaller sets and for each of the k folds a model is trained using $k-1$ of the folds as training data. The resulting model is validated on the remaining part of the data (i.e., it is used as a test set to compute performance measures such as accuracy).

Due to the nature of the sketch data sets considered in the experiments, HBF49 utilizes different partitioning methods per data set (some data sets have predefined partitions, others have no writer information, etc.). For CVCsymb, HHRco, LaViola, and Sign a writer-independent (WI) CV scheme is adopted, where each CV fold contains the data from one writer. The global WI recognition rate is then averaged from k CV experiments, where k is the number of writers [25]. For some data sets a predefined partitioning into train, validation, and evaluation samples is provided. On LaViola, a predefined train/evaluation (T/E) partition is available on the data of each writer, which is used for writer-dependent (WD) performance evaluation. On CVCsymb, HHRco, and Sign a 10-fold CV (10-CV) is run on the data from each writer and then the performance is averaged over all the writers [25]. For IMISketch a 5-CV scheme is applied (as suggested by the dataset partitioning) and for Ironoff-digits a 10-CV scheme is chosen for measuring WI performance. In the case of NicIcon a predefined partitioning defines the CV scheme, while ILG is only tested in WD mode, as each user was allowed to define custom gestures. All sketches are normalized in terms of scale and offset as described by HBF49, with a spatial re-sampling of the sketches to a dimension of 128 [25]. Missing values are replaced by imputed mean values, and resulting feature vectors are normalized using the mean and standard deviation.

Due to minor differences in the obtained data sets, as discussed in section 4.1, and the differences in the choice of libraries, feature implementation details and pre-processing, the original HBF49 results cannot be used directly for comparison. The problem is that the resulting performance measurements differ from the absolute values of the original publication. Instead, the experiment is repeated in its original form, but using the above described implementation. The SVM classifier is set to the same parameters as in HBF49, with a Gaussian kernel, a *gamma* of 0.01 and a *C* of 100. In contrast, the 1NN classifier has no additional parameters.

To summarize, the experiment in this section evaluates the classification accuracy of two classifiers (NN vs. SVM), on each of the data sets (and their partitioning), depending on the input feature set (49 vs. 165 features). During the first run, only the 49 features of the HBF49 feature set are used. The resulting measurements are then compared to a second run in which all 165 features are used as input. To test the hypothesis, that more features provide a better recognition accuracy, a paired-samples t-test is conducted to test for a significant difference in performance measurements.

5.1.2 Results

The results of both experiment runs are visualized in figure 5.1 and the measurements are summarized in table 5.1. By using all 165 features for training the classifiers the recognition accuracy improves for all data sets and conditions by up to +2.5%, except for CVCsymb in WI-CV conditions and IMISketch with SVM classification. On average the NN classification accuracy improves by +0.9% and SVM accuracy improves by +1.2%, with the highest impact for NN on NicIcon (WI T+V/E) of +2.4%, and +2.5% for the SVM approach on HHReco (WI-CV). The absolute recognition accuracy values reach 94.7% for NN and 96.0% for SVM. In comparison, the HBF49 benchmark results with only 49 features as input reach an average of 93.8% accuracy for the NN and 94.8% accuracy for the SVM approach. A slight decrease in accuracy is measured for both NN (-0.3%) and SVM (-0.8%) on the CVCsymb data set (WI-CV), and on the IMISketch SVM with -0.1%.

In order to compare whether the improvements in accuracy have a significant impact, a two-tailed paired samples t-test is conducted on the individual measurements. The comparison of NN classification accuracies shows a significant difference ($M = 0.91$, $SD = 6.68$; $t(12) = 4.39$, $p = .001$). Similarly, the comparison of SVM classification accuracies shows a significant difference as well ($M = 1.2$, $SD = 10.04$; $t(12) = 4.73$, $p < .001$).

5.1.3 Discussion

The hypothesis that the set of 165 digital pen features provides a significantly higher recognition accuracy than the 49 features of HBF49 is accepted due to the results of the t-test. It was possible to improve almost all measures, except CVCsymb and IMISketch. The difference for the latter one with -0.1% is almost negligible and could be the results of several random factors, including the exact split of the CV folds. In turn, the difference in the CVCsymb data set is a bit higher. The data set consists of 25 classes of architectural drawings from 12 subjects, without timestamp and pressure information. Examining the samples closer it appears that there is some variance in the way different subjects drew certain symbols. The larger feature set may be more sensitive to the WD aspects of the sketches. On the other hand, this assumption can be discarded as the other data sets also contain WD and WI modes and do not show this effect. Certain aspects of the ML experiments rely on the generation of random numbers, which includes the

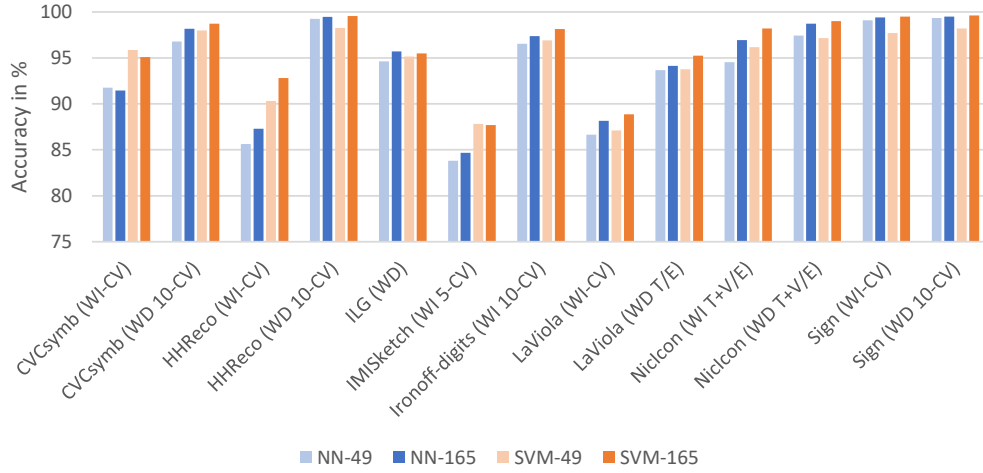


Figure 5.1: Visualization of classification accuracies for the HBF49 benchmark experiment. NN-49 & SVM-49 are the original classifiers trained on the 49 features of HBF49, while NN-165 & SVM-165 are trained on all 165 features presented in this thesis. NN = Nearest Neighbor, SVM = Support Vector Machine, WI = writer-independent, WD = writer-dependent, CV = cross-validation, T/V/E = predefined data set split into training (T), validation (V) and evaluation (E) data based on the HBF49 benchmark experiment design.

Data set	Experiment	NN-49	NN-165	SVM-49	SVM-165
CVCsymb	WI-CV	91.76%	91.45% (-0.3)	95.85%	95.10% (-0.8)
	WD 10-CV	96.79%	98.17% (+1.4)	97.99%	98.73% (+0.7)
HHReco	WI-CV	85.64%	87.30% (+1.7)	90.32%	92.81% (+2.5)
	WD 10-CV	99.25%	99.47% (+0.2)	98.27%	99.55% (+1.3)
ILG	WD	94.64%	95.70% (+1.1)	95.14%	95.49% (+0.3)
IMISketch	WI 5-CV	83.80%	84.66% (+0.9)	87.81%	87.71% (-0.1)
Ironoff-digits	WI 10-CV	96.55%	97.36% (+0.8)	96.91%	98.14% (+1.2)
LaViola	WI-CV	86.65%	88.15% (+1.5)	87.11%	88.88% (+1.8)
	WD T/E	93.66%	94.15% (+0.5)	93.77%	95.25% (+1.5)
Niclcon	WI T+V/E	94.54%	96.95% (+2.4)	96.16%	98.20% (+2.0)
	WD T+V/E	97.44%	98.73% (+1.3)	97.14%	98.99% (+1.9)
Sign	WI-CV	99.11%	99.39% (+0.3)	97.70%	99.49% (+1.8)
	WD 10-CV	99.34%	99.49% (+0.2)	98.19%	99.63% (+1.4)
		93.78%	94.69% (+0.9)	94.80%	96.00% (+1.2)

Table 5.1: Comparison between the HBF49 benchmark classifiers (NN-49 & SVM-49) with 49 features as input against the set of 165 features introduced in this thesis (NN-165 & SVM-165). NN = Nearest Neighbor, SVM = Support Vector Machine, WI = writer-independent, WD = writer-dependent, CV = cross-validation, T/V/E = predefined data set split into training (T), validation (V) and evaluation (E) data based on the HBF49 benchmark experiment design.

function for splitting data as part of the CV and the SVM classifier itself. A fixed value is used to initialize the pseudo-random number generation, to produce the same results on multiple runs. A quick test reveals that this parameter indeed can have an impact on the final recognition accuracy. To avoid over-fitting the value is left at the default, which unfortunately leads to slightly worse recognition results for the CVCsymb data set.

A common approach in pre-processing feature vectors for ML algorithms is to replace missing values and normalize the feature vectors before training. Missing values can occur for a number of reasons, in this case, not all data sets include timestamp and pressure information, so all features that rely on this input cannot be calculated. It can also happen that certain values cannot be calculated for certain samples, e.g., angles for overlapping points. Unfortunately, the HBF49 benchmark does not describe which of pre-processing steps were taken on the calculated feature vectors. Also, the number of samples and classes in the data sets as described in HBF49 varies from the statistics in the official data set publications and the downloaded data sets used here. This could indicate that the authors of HBF49 left out certain samples or only considered a subset. For the above reasons therefore the results of the experiments were not directly compared to the published results in HBF49 [25]. Instead, the experiment with the 49 features was repeated as closely as possible and then compared to the set of 165 features.

5.2 Comparison of ML Methods for Sketch Recognition

With the previous experiment results showing that using more digital pen features can have a positive impact on classification accuracy, the next experiment tests the performance of ten popular off-the-shelf supervised ML algorithms trained on all 165 features. The goal is to investigate which types of ML algorithms work best on a wide variety of data sets from different domains. For this experiment, the eight data sets from the HBF49 benchmark are reused. In addition, the DFKI-symbols data set and the How Humans Sketch data set are added to the selection. In the context of neurocognitive testing, the DFKI-symbols data set represents symbols, shapes and gestures which are commonly found in cognitive assessments, while the How Humans Sketch data set contains 250 classes with sketches from 1,350 unique participants. With a total of almost 20,000 sketches, it is a very complex real-world data set worth including into the selection. Overall, the selection of ML methods and data sets is motivated by the aim to create a baseline for future feature-based sketch recognition experiments, and to guide researchers in the selection of ML algorithms for particular sketch recognition tasks.

5.2.1 Methodology

In this experiment ten of the most common ML classifiers are compared against each other. These include linear SVMs, Gaussian SVMs, Logistic Regression, Nearest Neighbors, Naive Bayes, Decision Trees, Random Forests, AdaBoost, Gradient Boosted Trees and Deep Learning. Details about these algorithms are provided in section 2.2 - *Machine Learning* on page 5. The HBF49 benchmark from the previous experiment selects very specific hyperparameters for the ML classifiers and custom data splits for training and test data. Although this approach may result in improved recognition accuracies, it also bears the risk of selecting a combination that is a local optimum (overfitting). To provide a more generalized approach, no specific train test split is used in this experiment and all hyperparameters are set to their default values. While this may lead to lower recognition

accuracies, it also provides a more generalizable and unbiased view on the classifiers' out-of-the-box recognition performance. In turn, the impact of hyperparameter optimization is explored as part of the next experiment in section 5.3.

As in the previous experiment the scikit-learn library is used as the ML backend. For the linear SVM, the LIBSVM version is used with a linear kernel, a C parameter of 1.0 and a scaling *gamma* that is dependent on the number of input features. Similarly, the RBF kernel is used with the same parameters for the Gaussian SVM. The Logistic Regression is set to the default solver *lbfgs*, with *l2* penalty and a C of 1.0. A 5-Nearest Neighbors approach is used with uniform weights, a leaf size of 30, and Euclidean distance (*l2*) for the Minkowski metric. The only hyperparameter of Naive Bayes, the variance smoothing is set to 10^{-9} . For Decision Trees, the Gini impurity is used with best split selection and an unlimited number of leaf nodes. The Random Forests approach is set to the same values with bootstrap enabled. A Decision Tree is used as the base estimator for the AdaBoost algorithm, with a *learning rate* of 1 and the *SAMME.R* real boosting algorithm. The Gradient Boosted Trees are set to deviance (*logistic regression*) for classification with probabilistic outputs, a *learning rate* of 0.1, 100 estimators and the *FriedmanMSE* criterion. A multi-layer perceptron is used as a feedforward Deep Learning network with an *alpha* of 1, 1000 hidden layers, a learning rate of 0.001, *ReLU* activation and *Adam* optimizer.

As in the previous experiment, the sketches are first normalized in terms of scale and offset, before all 165 features are used to calculate the feature vectors. Normalization in this context means that each sketch is first translated to the origin of the Cartesian coordinate system (0, 0) and then scaled to a maximum x/y of 1.0 while keeping the aspect ratio. This ensures that no information from the sensor input is lost and all feature functions produce comparable results. The scaling and resampling of sketches to a dimension of 128, as in HBF49, is not performed, as it is an uncommon practice that is not motivated nor explained by the authors of the HBF49 publication [25]. Writer information is removed from the input to create equal conditions between all data sets. A randomized 10-fold CV scheme is adopted for each pair of ML method and data set, and the accuracy is averaged over the ten executions.

5.2.2 Results

The averaged classification accuracies are visualized in figure 5.2. Overall, the best results are achieved by Deep Learning (89.8%), followed by Random Forests (89.5%) and Gaussian SVMs (89%). Naive Bayes, Decision Trees and AdaBoost perform worst with just above 80% average accuracy. A more detailed summary of the results is provided in table 5.2. It shows the top 5 ML classifiers per data set sorted by accuracy and includes precision, recall, and the F1 score. Precision is the fraction of classification predictions produced by the model that were correct. The number displayed here is the micro-average precision across all classes. Similarly, recall is the fraction of classification labels that were successfully predicted by the model, which is also micro-averaged across all classes. The F1 score is the harmonic mean of precision and recall. It is a useful metric when looking for a balance between precision and recall for uneven class distributions. For all three metrics, the best possible value is 1.0 or 100%.

Based on the results in table 5.2, the prediction performance for the CVCsymb data set, which includes 25 types of symbols from 4,278 architectural drawings, is exceptionally good, with the top 2 classifiers reaching 100% accuracy. HHReco (13 classes/7,410 samples), NicIcon (14 classes/24,441 samples) and Sign (17 classes/33,154) reach close to perfect performance as well. Performance metrics on Ironoff-digits (10 classes/4,086

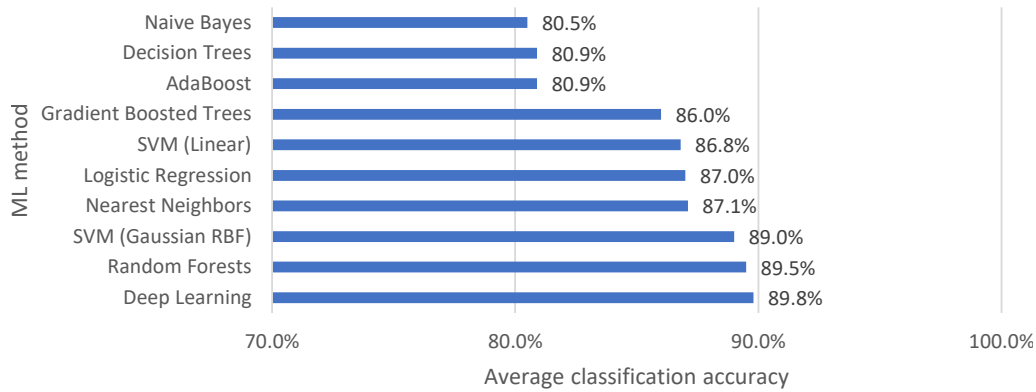


Figure 5.2: Sketch recognition classification accuracies averaged over all ten data sets.

samples) and DFKI-symbols (11 classes/9,938 samples) are around 98% for the top 5 classifiers, while LaViola (51 classes/10,655 samples) and IMISketch (13 classes/1,871 samples) are at around 94% and 90% accuracy respectively. The average accuracy for the writer-dependent ILG data set (21 classes/4,704 samples) is at 80% and less. Only the How Humans Sketch data set (250 classes/19,995 samples) is notably below chance level. Precision and recall scores are on par with classification accuracy. High scores for both show that the classifier is returning accurate results (high precision), as well as returning a majority of all positive results (high recall). The lower-performing classifiers on data sets like IMISketch, ILG and How Humans Sketch have fittingly lower values. Noticeably, the results for NN and SVM differ from the HBF49 benchmark, due to the writer-independent data split and the inclusion of two additional data sets.

Table 5.2 also includes the area under the curve (AUC) for the receiver operating characteristic (ROC) curves in figures 5.3 and 5.4. ROC curves feature the true positive rate on the y axis and the false positive rate on the x axis. This means that the top left corner of the plot is the "ideal" point, it has a false positive rate of zero and a true positive rate of one. This is not very realistic for most examples, but it does mean that a larger AUC is usually better. The steepness of ROC curves is also important since it is ideal to maximize the true positive rate while minimizing the false positive rate. Classifiers like the Gaussian SVMs and Deep Learning produce rather steep curves with high AUC measures for all ten data sets, while algorithms such as AdaBoost, which on average perform worse, also produce flatter ROC curves. Furthermore, it is noticeable that all models struggle with the prediction of the ILG and How Humans Sketch data sets, as can be seen by the lower ROC curves and smaller AUC values.

5.2.3 Discussion

Considering the popularity and recent advances in deep learning it is noteworthy that even a simple feedforward network, such as used in this case, can be trained to make predictions with high accuracy. In general, this process requires a lot of training data, but as can be observed from the results even small data sets, like IMISketch with less than 2,000 samples can be used to train models achieving accuracies of almost 90% with solid precision and recall. Same as all other methods deep learning still struggles with the Sign and the How Humans Sketch data set judging from the performance metrics and ROC curves. For How Humans Sketch it produces the highest measures amongst

Data set	ML method	Accuracy	Precision	Recall	F1 score	AUC (ROC)
CVCsymb	Logistic Regression	100.0%	100.0%	100.0%	1.000	1.000
	SVM (Linear)	100.0%	100.0%	100.0%	1.000	1.000
	Deep Learning	99.9%	99.9%	99.9%	0.999	1.000
	SVM (Gaussian RBF)	99.8%	99.8%	99.8%	0.998	1.000
	Random Forest	99.2%	99.2%	99.2%	0.992	1.000
HHReco	Random Forest	99.8%	99.8%	99.8%	0.998	1.000
	SVM (Gaussian RBF)	99.4%	99.4%	99.4%	0.994	1.000
	SVM (Linear)	99.2%	99.2%	99.3%	0.993	0.999
	Gradient Boosted Tree	99.2%	99.2%	99.2%	0.992	1.000
	Nearest Neighbors	99.2%	99.2%	99.2%	0.992	0.995
ILG	Nearest Neighbors	80.7%	80.4%	80.2%	0.801	0.898
	Random Forest	79.4%	79.7%	78.4%	0.788	0.972
	Deep Learning	75.9%	75.9%	74.7%	0.749	0.970
	SVM (Gaussian RBF)	73.3%	73.3%	71.3%	0.717	0.966
	Gradient Boosted Tree	65.7%	65.3%	63.3%	0.635	0.939
IMISketch	Random Forest	91.5%	90.7%	84.7%	0.865	0.993
	Gradient Boosted Tree	91.2%	90.2%	82.7%	0.851	0.989
	Logistic Regression	89.1%	87.5%	81.6%	0.838	0.989
	Deep Learning	88.8%	86.7%	83.9%	0.848	0.993
	SVM (Gaussian RBF)	88.8%	90.3%	77.8%	0.820	0.988
Ironoff-digits	Deep Learning	98.5%	98.6%	98.5%	0.986	0.999
	Logistic Regression	98.5%	98.6%	98.5%	0.986	0.999
	SVM (Linear)	98.2%	98.2%	98.2%	0.982	0.997
	Gradient Boosted Tree	97.7%	97.8%	97.6%	0.977	0.999
	SVM (Gaussian RBF)	97.7%	97.8%	97.6%	0.977	0.999
LaViola	Logistic Regression	94.5%	93.7%	93.6%	0.936	0.999
	SVM (Gaussian RBF)	94.2%	92.8%	92.7%	0.926	0.998
	SVM (Linear)	93.8%	92.5%	93.0%	0.925	0.998
	Random Forest	93.7%	92.4%	92.2%	0.922	0.998
	Deep Learning	93.6%	94.5%	92.6%	0.928	0.999
NicIcon	SVM (Gaussian RBF)	99.5%	99.5%	99.5%	0.995	1.000
	Logistic Regression	99.4%	99.4%	99.4%	0.994	1.000
	SVM (Linear)	99.3%	99.3%	99.3%	0.993	1.000
	Nearest Neighbors	99.2%	99.2%	99.2%	0.992	0.996
	Gradient Boosted Tree	99.1%	99.1%	99.1%	0.991	1.000
Sign	Logistic Regression	99.7%	99.7%	99.7%	0.997	1.000
	Random Forest	99.7%	99.7%	99.7%	0.997	1.000
	SVM (Linear)	99.7%	99.7%	99.7%	0.997	1.000
	Gradient Boosted Tree	99.7%	99.7%	99.7%	0.997	0.999
	Deep Learning	99.6%	99.6%	99.6%	0.996	1.000
DFKI-symbols	Random Forest	98.9%	99.0%	98.9%	0.989	1.000
	SVM (Linear)	98.9%	98.9%	98.9%	0.989	0.998
	Logistic Regression	98.8%	98.9%	98.9%	0.989	0.999
	Gradient Boosted Tree	98.8%	98.8%	98.8%	0.988	0.999
	Deep Learning	98.5%	98.6%	98.5%	0.985	0.999
HHS*	Deep Learning	44.7%	46.9%	44.7%	0.430	0.938
	Logistic Regression	42.4%	42.1%	42.8%	0.414	0.964
	SVM (Linear)	41.7%	40.3%	42.0%	0.397	0.952
	SVM (Gaussian RBF)	39.5%	40.9%	40.2%	0.390	0.950
	Random Forest	36.7%	36.4%	37.2%	0.350	0.882

Table 5.2: Top 5 sketch recognition classifiers per data set (*HHS = How Humans Sketch). The reported measures are average values (10-CV).

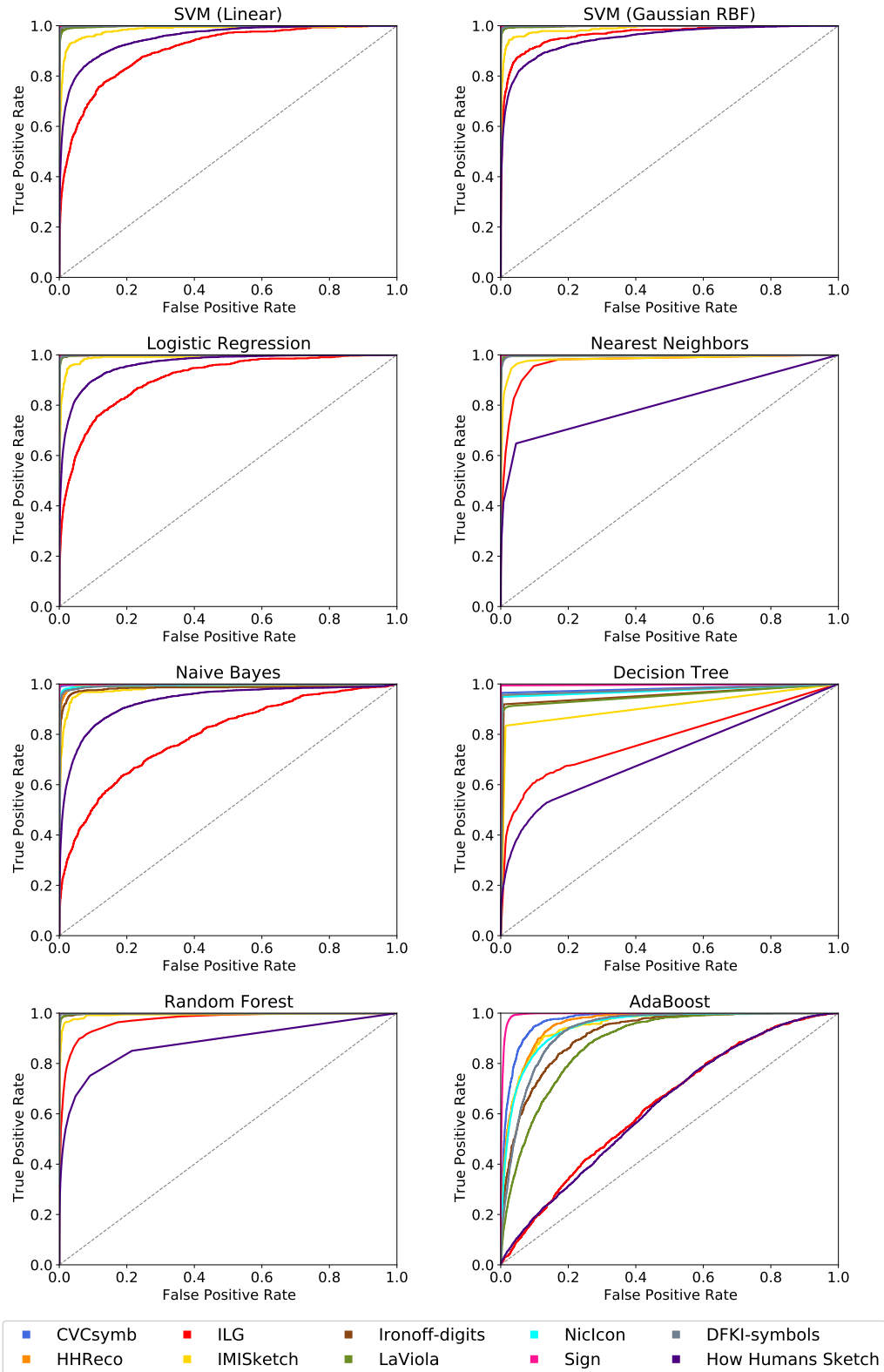


Figure 5.3: ROC curves for sketch recognition classifiers per ML method.

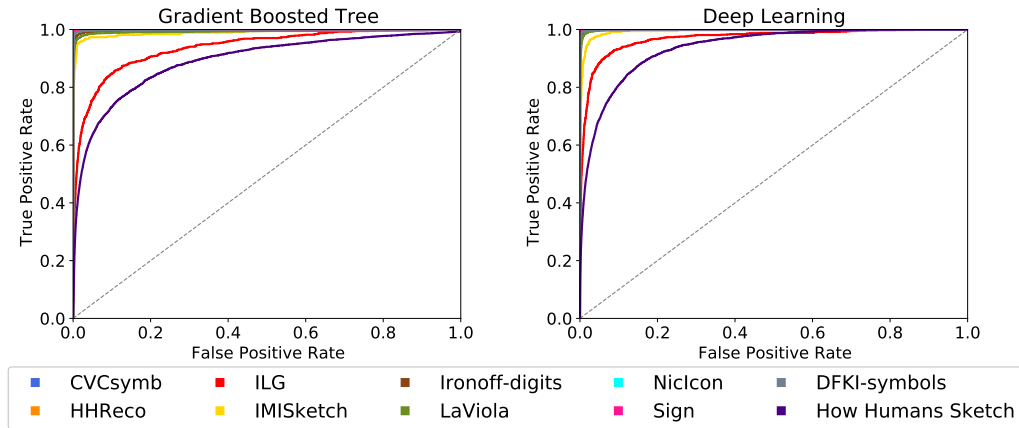


Figure 5.4: ROC curves for sketch recognition classifiers per ML method (continued).

the tested ML algorithms, but with 44.7% accuracy, it stays below chance level. Part of the reason can be explained with the small fraction of samples per class, which for How Humans Sketch is about 80 samples per class, while other data sets like DFKI-symbols have around 900 samples per class. Deep learning networks are susceptible to this, as all weights of the hidden layers need to be adjusted per class to make an accurate prediction.

The second-best average accuracy is achieved by Random Forests, which is also interesting since the other decision tree based approaches achieve considerably lower measures. Vanilla decision tree approaches achieve the lowest accuracy with 80.9%, which is to be expected, since the other variations are designed to avoid the potential pitfalls of decision trees. Accordingly, boosting improves their performance by 5% and the ensemble of decision trees used in the Random Forests approach achieves even better performance. In the scikit-learn framework used here, the AdaBoost approach uses a simple decision tree as the default estimator, which could be the reason why it does not measurably improve its accuracy. In general, decision tree based approaches benefit a lot from different data preparation techniques, such as feature selection, and hyperparameter tuning. The choice not to fine-tune any of the algorithm's parameters was made on purpose, to avoid overfitting and provide an unbiased view on the classifiers. However, preliminary testing showed a clear improvement of accuracy measures for different hyperparameters in decision tree based approaches. The effect of hyperparameter optimization is therefore explored in the next experiment. The Nearest Neighbors approach falls into the same category, as its performance relies heavily upon choosing the right k for the data at hand. But, as in the HBF49 benchmark, it still performs rather well.

Naive Bayes is an extremely simple classifier and it is not unexpected that it produces rather weak results. Its Gaussian estimators are probably very good (as can be seen for the SVM approach), but the naive assumptions are the problem (e.g., that all the variables in the data set are not correlated to each other). As such, the Naive Bayes approach can be viewed as the baseline accuracy for the other, more sophisticated classifiers.

Logistic regression and the linear SVM produce very similar results, which can be attributed to the basic concept behind these approaches, which is to separate the data linearly. As shown this can work well for many data sets, but it can be improved by reducing the overall feature space, e.g., by applying feature selection or dimensionality reduction techniques. The Gaussian SVM with RBF kernel solves this problem by design

through an embedded mapping of the input vectors into other dimensions. This might be one of the reasons why the Gaussian SVM consistently produces better results for the high dimensional input of 165 features compared to the linear approaches. Notably, SVMs belong to the category of parameterized ML classifiers and are therefore likely to be susceptible to hyperparameter optimization.

Overall, the off-the-shelf ML classifiers perform very well using the presented feature set on a variety of different data sets. The lower-performing data sets ILG and IMISketch are special cases, because the ILG set contains user-defined gestures, whose sketch characteristics cannot be generalized between different users, and the IMISketch data set, with its 1,800 samples is the smallest data set in the selection. Similarly, the How Humans Sketch presents a challenge for the feature-based classifiers, because of its high number of classes and low ratio of samples per class.

5.3 AutoML for Sketch Recognition

The first experiment showed that using more digital pen features can produce better results than using fewer features. The second experiment then demonstrated that off-the-shelf ML methods with default settings can recognize sketches from various data sets with varying degrees of accuracy. In this last sketch recognition experiment, AutoML is utilized to automatically build and fine-tune state-of-the-art ML models with optimized hyperparameters. The experiment aims to show that feature-based ML methods for sketch recognition can be optimized on individual data sets to produce better results.

5.3.1 Methodology

Google AutoML Tables⁸ is used as the framework to conduct the experiments. It is a cloud-based, codeless end-to-end ML platform that provides common feature engineering tasks, such as normalization of numeric features, and one-hot encoding and embeddings for categorical features. First, the feature vectors for each sketch of each data set are calculated offline using the digital pen features library presented in chapter 3. The resulting data is then exported and uploaded into the cloud. A regional server selection allows compliance with European data privacy laws. After selecting the target column and input features, AutoML Tables pre-processes the data set and starts training for multiple model architectures at the same time. This approach enables AutoML Tables to determine the best model architecture quickly, without having to serially iterate over the many possible model architectures, which is one of the reasons why it was selected for the scope of this thesis. Training the same amount of models and performing hyperparameter optimization on them using a standard issue GPU desktop machine is time-consuming and therefore not feasible. Other approaches, such as RapidMiner [61] and auto-sklearn [31, 32] were tested as well. RapidMiner struggled to produce better results than presented in the previous experiment and provided very little configuration options, while auto-sklearn did not converge even after letting it run overnight on a high-performance desktop machine. Because of this, Google AutoML Tables was selected, which includes several model architectures, including linear models, feedforward deep neural networks, gradient boosted decision trees, AdaNet and ensembles of various model architectures.

⁸<https://cloud.google.com/automl-tables>

For each data set, the feature vectors are calculated in the same way as in the previous experiments. The AutoML training is started on each data set individually with all 165 features selected as input. A default automatic split of 80% training data, 10% validation data and 10% test data is selected. Since the framework is currently in the beta phase the only optimization criterion available is to optimize for log loss. Log loss is the cross-entropy between the model predictions and the label values. This ranges from zero to infinity, where a lower value indicates a higher-quality model. Each model training is started with a budget of one node hour per model, which is the recommended value for data sets with less than 100,000 rows of data. A node hour describes a unit of computing time on the cloud computing cluster. Early stopping is enabled for model training, which ends model training when it detects that no more improvements can be made.

5.3.2 Results

Unfortunately, in the current state of the framework, the choice of options for exporting the model is very limited, which is why the following figures and tables differ from previous experiments. Table 5.3 summarizes the results of the AutoML experiments, including the accuracy averaged over all classes, the F1 score, the log loss, precision, recall and the AUC values of ROC curves and precision-recall (PR) curves. The corresponding curves are visualized in figures 5.5 and 5.6. The precision-recall curve shows the trade-off between precision and recall at different classification score thresholds. A lower score threshold results in higher recall but typically lower precision, while a higher threshold results in lower recall but typically higher precision. Similar to the ROC AUC, the area under the precision-recall curve ranges from zero to one, where a higher value indicates a higher-quality model.

The results in table 5.3 show very promising model performance for almost all data sets, except How Humans Draw. Several models produce near-perfect results. Log loss values of below 0.1 indicate high-quality models for CVCsymb, HHReco, Ironoff-digits, NicIcon, Sign and DFKI-symbols, while LaViola still achieves 0.149 with solid precision, recall and AUC values. The IMISketch results are a bit worse with a log loss of 0.3 and precision and recall values of about 90%. The writer-dependent ILG data set still achieves an accuracy of 98.3% with 88.8% precision and only 74.4% recall. Despite the 99.7% accuracy, the log loss of the How Humans Sketch data set is very high with 2.09 and a very low F1 score, recall and PR AUC, indicating a bad model quality. This is further supported by the appearance of the corresponding curves in figure 5.6. In contrast, the higher quality models produce near-perfect ROC and PR curves with high AUC values.

5.3.3 Discussion

The AutoML approach achieves superior recognition performance on all data sets except one. How Humans Sketch is an excellent example of a deceptive accuracy measure, if considered standalone without its corresponding precision and recall values. The recall reveals that the number of truly relevant results is very low. A very high log loss value in the context of AutoML indicates that the optimization process was suboptimal during training the model. To rule out that the model training was stopped too early, the experiment was repeated using a higher budget for training. Nevertheless, the results stayed the same. This might indicate that the How Humans Sketch data set is not well suited as input for feature-based, supervised ML algorithms.

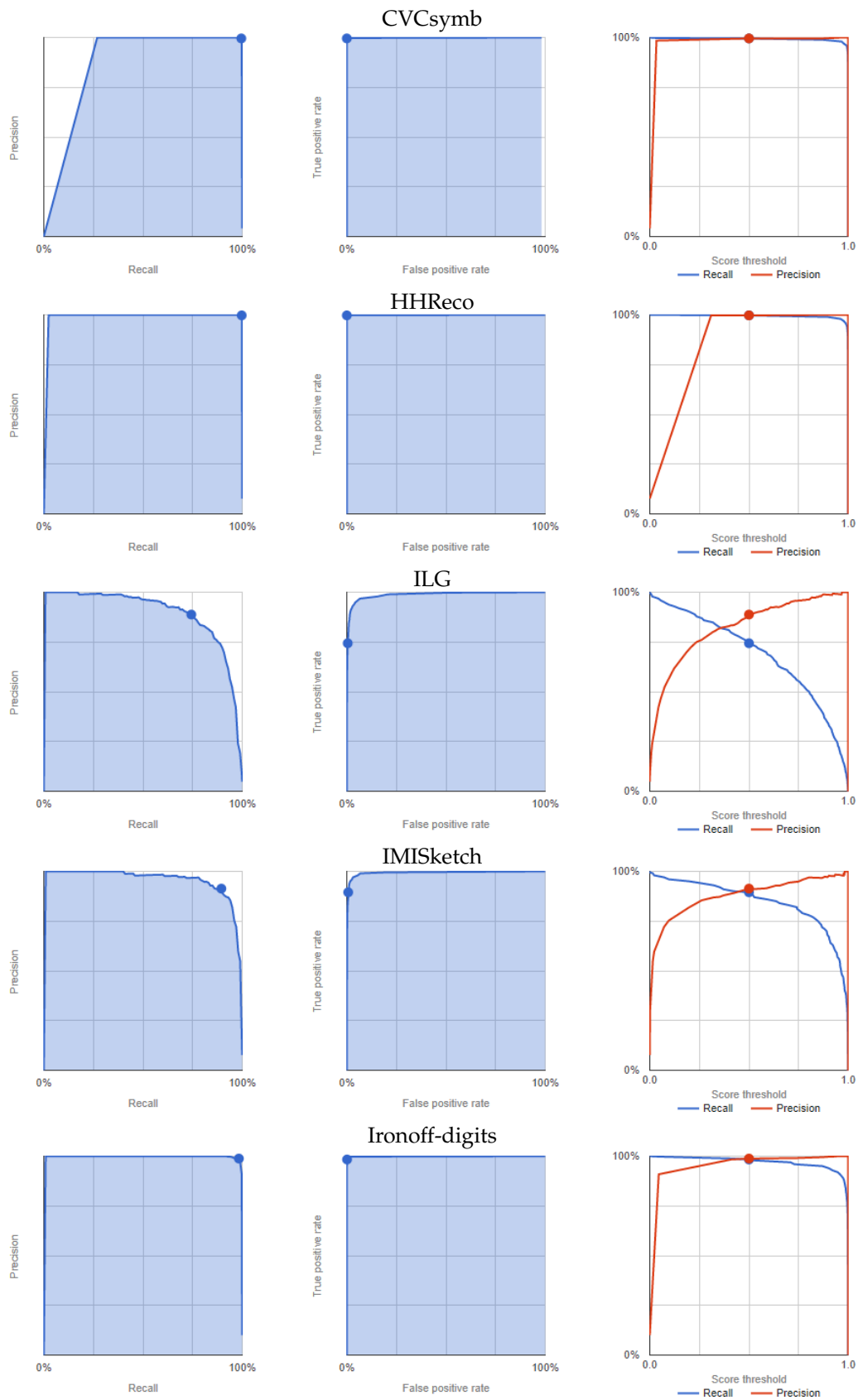


Figure 5.5: Precision-recall curves (left), ROC curves (middle) and precision-recall score thresholds (right) for AutoML experiments. Dots show the 0.5 score threshold.

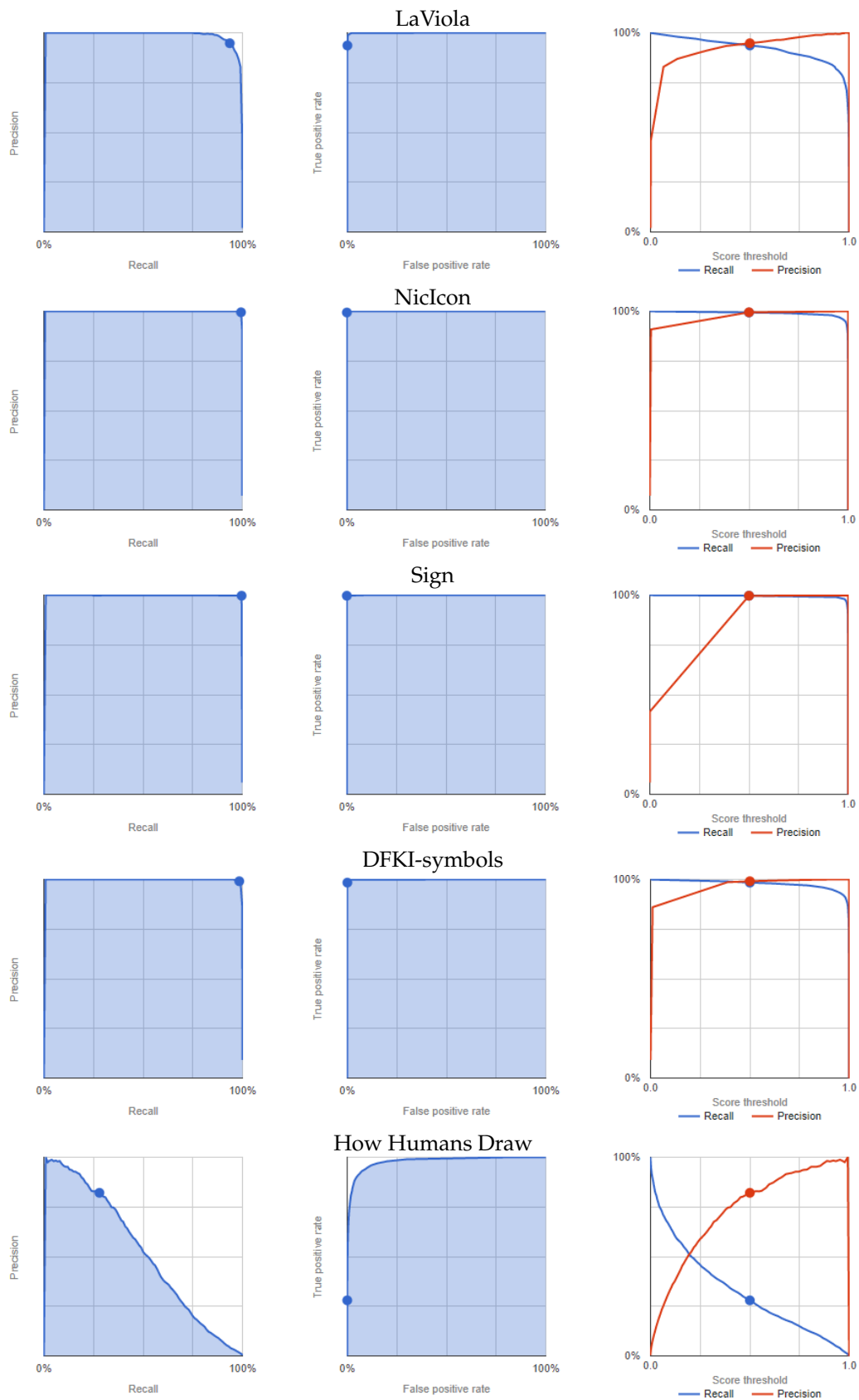


Figure 5.6: Precision-recall curves (left), ROC curves (middle) and precision-recall score thresholds (right) for AutoML experiment results (continued).

Data Set	Accuracy	F1 score	Log loss	Precision	Recall	AUC (PR)	AUC (ROC)
CVCsymb	99.9%	0.995	0.015	99.5%	99.5%	0.999	0.999
HHReco	99.9%	0.997	0.006	99.7%	99.7%	0.999	0.999
ILG	98.3%	0.809	0.603	88.8%	74.4%	0.896	0.988
IMISketch	98.5%	0.904	0.300	91.3%	89.5%	0.958	0.993
Ironoff-digits	99.7%	0.986	0.048	98.8%	98.3%	0.998	0.999
LaViola	99.7%	0.942	0.149	94.7%	93.6%	0.988	0.999
NicIcon	99.9%	0.994	0.017	99.5%	99.4%	0.999	1.000
Sign	99.9%	0.997	0.024	99.7%	99.7%	0.998	1.000
DFKI-symbols	99.7%	0.987	0.037	99.1%	98.4%	0.999	0.999
How Humans Sketch	99.7%	0.417	2.090	82.1%	28.0%	0.523	0.979

Table 5.3: Results of the AutoML experiments for sketch recognition with all 165 digital pen features as input. Data split: 80% training, 10% validation and 10% testing.

Notably, the accuracy on the writer-dependent ILG data set and the small IMISketch data set improves considerably in comparison with the off-the-shelf ML classification of the last experiment. Although the log loss for ILG is higher than for the other data sets, all other metrics, including precision, recall and AUC values are satisfactory. Even for the small IMISketch data set AutoML achieves high performance metrics and improves recognition accuracy by 7% compared to the unoptimized ML classifiers.

From a user perspective, the AutoML approach drastically reduces the human effort necessary for applying machine learning. The input is provided as a comma-separated values (CSV) file, where each row represents the feature vector of one sketch. After uploading the file using the web interface, a model training can be started by the push of a button, without writing a single line of code. Unfortunately, the number of options and parameters available for modifying the training process is very limited. Users can allocate a certain amount of computing resources for training and receive an email once the model has finished training. A limited overview of the performance measurements is provided in the web interface together with the curve visualizations. However, no export function for these metrics is provided, nor is there any additional explanation about how the underlying trained model looks like. The model can be exported within the Google cloud infrastructure to run online and batch predictions at additional costs. Unfortunately, the only available download format for the models is a proprietary file format, which could not be closer examined with available tools.

5.4 ML-based Cognitive Performance Classification

The last set of experiments is concerned with the ML-based classification of cognitive test performance. Cognitive assessments are commonly used in medicine to either diagnose disease (e.g., dementia, Parkinson's, etc.) or to screen for signs of cognitive impairment, e.g., as part of large-scale community screening programs. In both use cases, the underlying task is to categorize subjects either as healthy or as suspicious, the latter of which would indicate to conduct additional testing or medical interventions. Traditional approaches for digitalizing cognitive assessments are discussed in sections

2.4 and 3.3. These methods are proven to be successful for a wide variety of cognitive assessments, but they are also highly dependent on the specific test and often analysis components include task-related semantic knowledge that needs to be modeled manually for each type of test. The next big step in analyzing digital cognitive assessments is to predict cognitive performance independently of the test content, by looking only at the writing and sketching behavior of users. This approach allows us to generalize the classification of cognitive performance independently of the considered task and opens up new approaches for transparent behavior analysis in pen-based intelligent user interfaces. Users do not longer have to perform specific tests, instead, their handwriting and sketch input can be analyzed transparently during common pen-based tasks like sketching, taking notes or writing messages. In the future, such a prediction of cognitive performance can be used in interactive pen-based systems to adapt task difficulty and content in real-time, depending on the cognitive state of the user. This is why the last experiment aims at predicting cognitive performance by only considering the syntactic digital pen features, which are applicable independent of the task, without performing further content analysis.

5.4.1 Methodology

For the experiments of this section, the cognitive assessments data set is used, which was collected as part of the Interakt project described in section 4.2. It contains a total of 236 sketch samples of the CDT, TMT and ROCF. The samples were manually scored and annotated by experts, who labeled each sample as either healthy or suspicious. In total 152 samples were classified as healthy and 84 as suspicious (14/40 CDT samples (35.0%), 14/80 TMT samples (17.5%), and 56/116 ROCF samples (48.3%)). The data was collected from 40 elderly subjects, who participated in a study at a geriatrics daycare clinic. More details on the subjects and the collection method are provided in section 4.2.2 - *CDT, TMT & ROCF Data Collection Method* on page 33.

Based on the task design all previous experiments considered only sketch-based digital pen features, meaning that the entire sketch was used as input for each of the feature functions to produce the feature vector. However, many features, especially all of Rubine's features [88] can be applied on a stroke-level as well. Taking into account stroke-level features as well serves a double purpose. Firstly, many gestures and symbols consist of only one stroke anyways and the research question arises, whether or not one can predict cognitive performance from the sketching behavior of as little data as single strokes, or if entire sketches are required. Secondly, due to unforeseen circumstances in regard to the global pandemic, the data collection had to be cut short, resulting in a rather small number of overall sketch samples, but ML approaches usually require a larger data set as input. In fact, several of the related sketch recognition publications in section 2.2 (*Machine Learning*) are focused on the stroke-level rather than entire sketches. Dividing the data set into labeled strokes results in a total of 6,893 strokes (508/1,211 CDT (41.9%), 117/438 TMT (26.7%), and 2,175/5,244 ROCF (41.5%)).

Three feature subsets are considered the set of all 165 features, a set of 11 features from related publications from the field of analyzing cognitive assessments, and the combination of both (176 features). The set of 11 features collected from literature that focuses on the evaluation of cognitive performance [21, 24, 107], and includes number of strokes, sketching time, stroke distance, duration, average pressure, average velocity, variation of velocity, number of pauses, average pause duration, the ratio between sketching and pausing, and average lift duration.

The set of experiments conducted here is concerned with the binary classification of samples as healthy or suspicious, on a combination of sketch-level vs. stroke-level features, the three feature subsets, and the same ten ML methods from above experiments. This results in a total of 60 experiment runs, where each classifier is trained individually on each of the data and feature sets in a 10-fold CV scheme. Simple iterative hyperparameter optimization is employed to find optimal settings for each of the ML classifiers. The SVMs are set to a C of 1.5 with a maximum number of iterations of 10,000. Logistic Regression is set to a C of 8.0 with *newton-cg* as solver. A 7-Nearest Neighbors approach is used, and Naive Bayes is set to a smoothing value of 10^{-9} . All decision tree based approaches are left at default settings, only the learning rate of AdaBoost is set to 0.5. The Deep Learning network is set to 1,000 hidden layers with an *alpha* of 1, a learning rate of 0.001, *Adam* as solver and *tanh* activation function.

In addition, AutoML experiments are conducted to verify whether the prediction accuracy can be improved for this problem as well. As in the previous experiment the Google AutoML Tables framework is used to conduct the experiment. Unfortunately, the approach requires at least 1,000 rows of data, which is why only the stroke-level conditions are considered for each of the feature subsets.

5.4.2 Results

The most accurate top 5 ML methods for the classification of cognitive test performance are summarized in table 5.4. A differentiation is made between sketch-based and stroke-based calculation of features. Sketch-176 includes all 165 sketch-based digital pen features, plus the 11 features related to cognitive testing, whereas Sketch-165 only includes all 165 features and Sketch-11 only includes the cognitive features. The Stroke-176, Stroke-165, and Stroke-11 are their equivalent stroke-based feature subsets. All reported measures are averaged as part of the applied 10-fold cross-validation scheme.

AdaBoost achieves the highest accuracy with 87.5% on the set of all 165 sketch-based features, followed by the linear SVM with 85.4%. A similar accuracy of 85.4% is achieved by Random Forests and the linear SVMs on the set of 176 features. Considering only the 11 features related to cognitive testing, the prediction accuracies drop below 80% with the highest accuracy achieved by the Deep Learning approach with 77.1%. Stroke-based approaches stay above the chance level with a maximum of 65.0% recognition accuracy for Gaussian RBF SVMs for all 176 features and 64.8% for the 165 features set. The corresponding ROC curves in figures 5.7 and 5.8 support this observation. All stroke-based approaches produce a flat ROC curve with a maximum AUC of 0.598 for the Gaussian RBF SVMs. In contrast, the ROC curves of their sketch-based equivalents are much steeper with the highest AUC achieved by the AdaBoost approach with a value of 0.877, followed by 0.861 for the linear SVMs.

The highest precision and recall scores are achieved by AdaBoost as well with 86.0% and 87.7% respectively. In general, the sketch-based feature sets with 165 and 176 features perform best, and stroke-level feature approaches are just above chance level. Similarly, the F1 scores are above 0.8 for the bigger sketch-level feature sets, which indicates a good balance between precision and recall. In contrast, the F1 scores for the stroke-level conditions are all below 0.6.

To help understand why the results are rather poor, a visualization of the underlying decision boundaries for the individual ML classifiers is provided in figure 5.9. The t-distributed stochastic neighbor embedding (t-SNE) algorithm presented by Van Der Maaten [103, 104] is used to map the high-dimensional feature space of 165 dimensions

Feature subset	ML method	Accuracy	Precision	Recall	F1 score	AUC (ROC)
Sketch-176	Random Forest	85.4%	85.5%	82.1%	0.833	0.821
	SVM (Linear)	85.4%	83.9%	86.1%	0.846	0.861
	Gradient Boosted Tree	83.3%	83.7%	79.1%	0.806	0.791
	AdaBoost	81.2%	79.7%	78.8%	0.792	0.788
	SVM (Gaussian RBF)	81.2%	81.9%	76.2%	0.778	0.762
Sketch-165	AdaBoost	87.5%	86.0%	87.7%	0.867	0.877
	SVM (Linear)	85.4%	83.9%	86.1%	0.846	0.861
	Gradient Boosted Tree	83.3%	83.7%	79.1%	0.806	0.791
	Random Forest	83.3%	82.4%	80.5%	0.812	0.805
	SVM (Gaussian RBF)	81.2%	81.9%	76.2%	0.778	0.762
Sketch-11	Deep Learning	77.1%	75.4%	73.0%	0.738	0.730
	Random Forest	77.1%	75.4%	73.0%	0.738	0.730
	AdaBoost	72.9%	71.9%	73.7%	0.719	0.737
	SVM (Gaussian RBF)	72.9%	71.8%	65.7%	0.665	0.657
	Decision Tree	70.8%	71.5%	73.4%	0.704	0.734
Stroke-176	SVM (Gaussian RBF)	65.0%	64.5%	59.8%	0.588	0.598
	Random Forest	63.7%	62.2%	59.2%	0.586	0.592
	Gradient Boosted Tree	63.5%	62.2%	58.3%	0.571	0.583
	AdaBoost	61.6%	59.5%	56.1%	0.542	0.561
	Logistic Regression	61.5%	59.1%	57.0%	0.563	0.570
Stroke-165	SVM (Gaussian RBF)	64.8%	64.4%	59.6%	0.585	0.596
	Gradient Boosted Tree	63.8%	62.5%	58.9%	0.579	0.589
	Random Forest	63.7%	62.5%	58.5%	0.573	0.585
	AdaBoost	61.8%	59.8%	56.1%	0.541	0.561
	SVM (Linear)	61.4%	59.0%	56.8%	0.560	0.568
Stroke-11	SVM (Linear)	61.6%	59.5%	55.7%	0.534	0.557
	Logistic Regression	61.3%	59.1%	55.6%	0.535	0.556
	Deep Learning	61.3%	59.2%	55.1%	0.523	0.551
	Gradient Boosted Tree	61.2%	58.7%	55.9%	0.544	0.559
	SVM (Gaussian RBF)	61.2%	59.3%	54.6%	0.511	0.546

Table 5.4: Top 5 ML methods for cognitive test performance classification per feature subset. The reported measures are average values (10-CV).

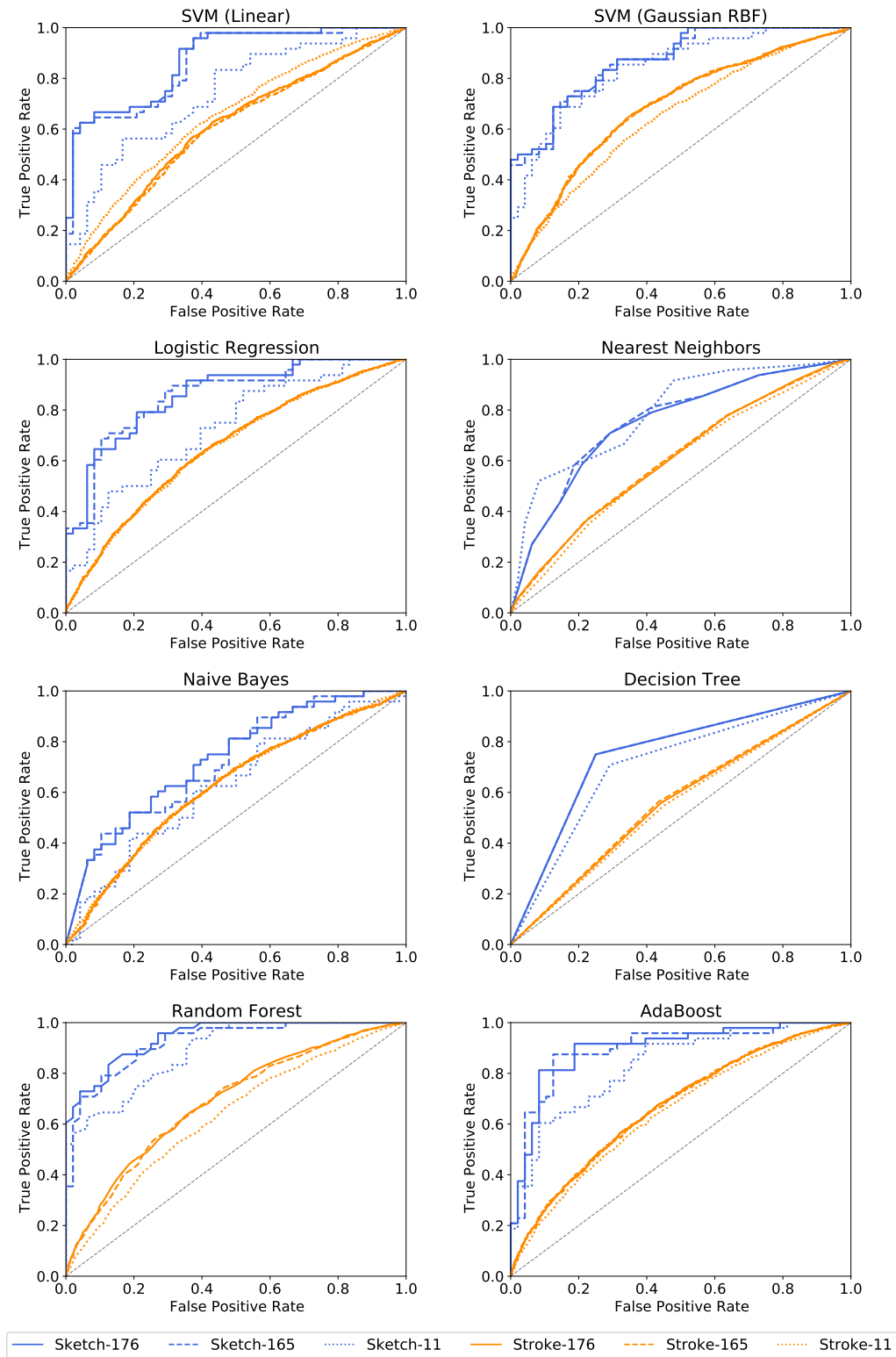


Figure 5.7: ROC curves for cognitive test performance classification per ML method and feature subset. The jagged curves are a result of the binary classification.

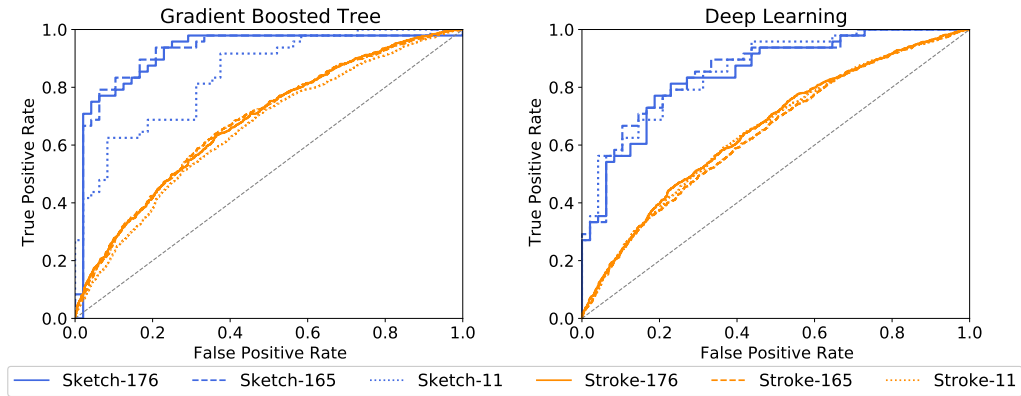


Figure 5.8: ROC curves for cognitive test performance classification per ML method and feature subset (continued).

(the digital pen features) down to the two-dimensional Euclidean space. In data sciences and ML, t-SNE is commonly used as a nonlinear dimensionality reduction technique, which is well-suited for embedding high-dimensional data for visualization in a low-dimensional space of two dimensions. Using standard techniques such as principal component analysis (PCA) or multi-dimensional scaling for dimensionality reduction results in crowded plots, where many data points fall close together in the mapped two-dimensional space, resulting in unhelpful layouts [29]. In contrast, t-SNE specifically addresses this problem by computing a mapping of distances in high-dimensional space to distances in low-dimensional space such that smaller pairwise distances in high-dimensional space (which would produce the crowding problem) are mapped to larger distances in two-dimensional space, while still preserving overall global distances [29]. The first two plots of figure 5.9 show the raw digital pen feature vectors for each sketch after being mapped using t-SNE (*components* set to 2, a *perplexity* of 30, *early exaggeration* set to 12.0 and a *learning rate* of 200.0 with a maximum of 1,000 iterations). Different classes are indicated by color (red = healthy, blue = suspicious), solid points are training samples, whereas semi-transparent points are used for testing. The Sign data set is included as a reference, because the experiments in section 5.2 show that its classes can be easily separated by the ML classifiers. The corresponding plot shows 17 almost completely separated clusters, one for each of the classes of the data set. All remaining plots of figure 5.9 show the decision boundaries of the ML classifiers for the prediction of cognitive test performance.

The results of the AutoML experiment on the stroke-level feature sets are summarized in table 5.5, with the corresponding precision-recall and ROC curves presented in figure 5.10. Due to the binary classification, the framework does not provide additional precision-recall score thresholds like for the previous AutoML experiment. The results show only a minor improvement in accuracy of 65.8%, with comparably high log loss and low precision, recall and F1 score. Similarly, the ROC curves are rather flat with the highest AUC ROC value of 0.671.

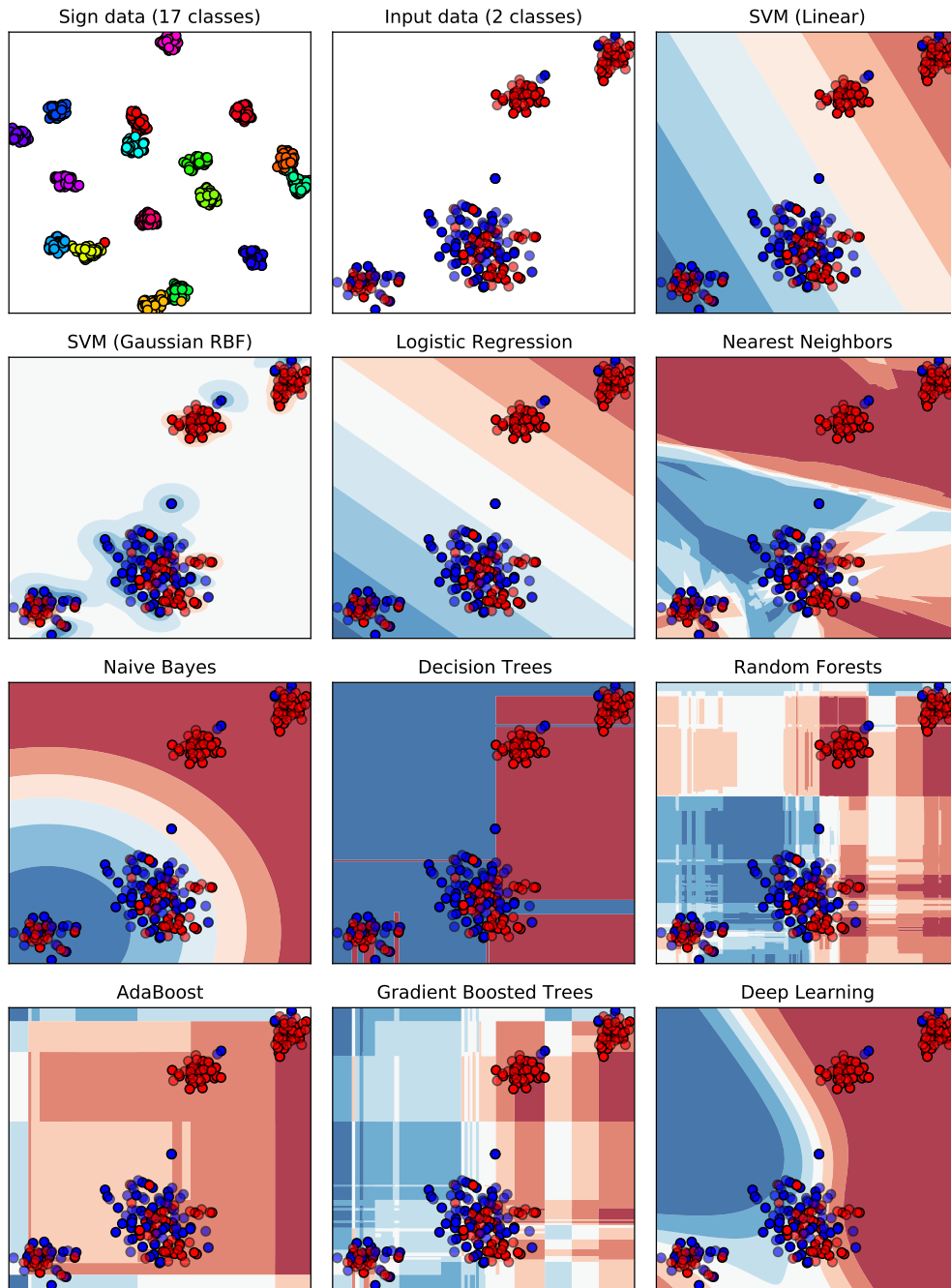


Figure 5.9: A visualization of decision boundaries for the ML classifiers. The feature space of 165 dimensions (features) is mapped to the 2d Euclidean space using t-SNE dimensionality reduction. Each dot represents the input feature vector of one sample from the data set. Classes are color-coded (red = healthy, blue = suspicious), solid points are training samples, while testing samples are semi-transparent. The Sign data set with its 17 classes is included as reference, to give an example of an easily separable data set.

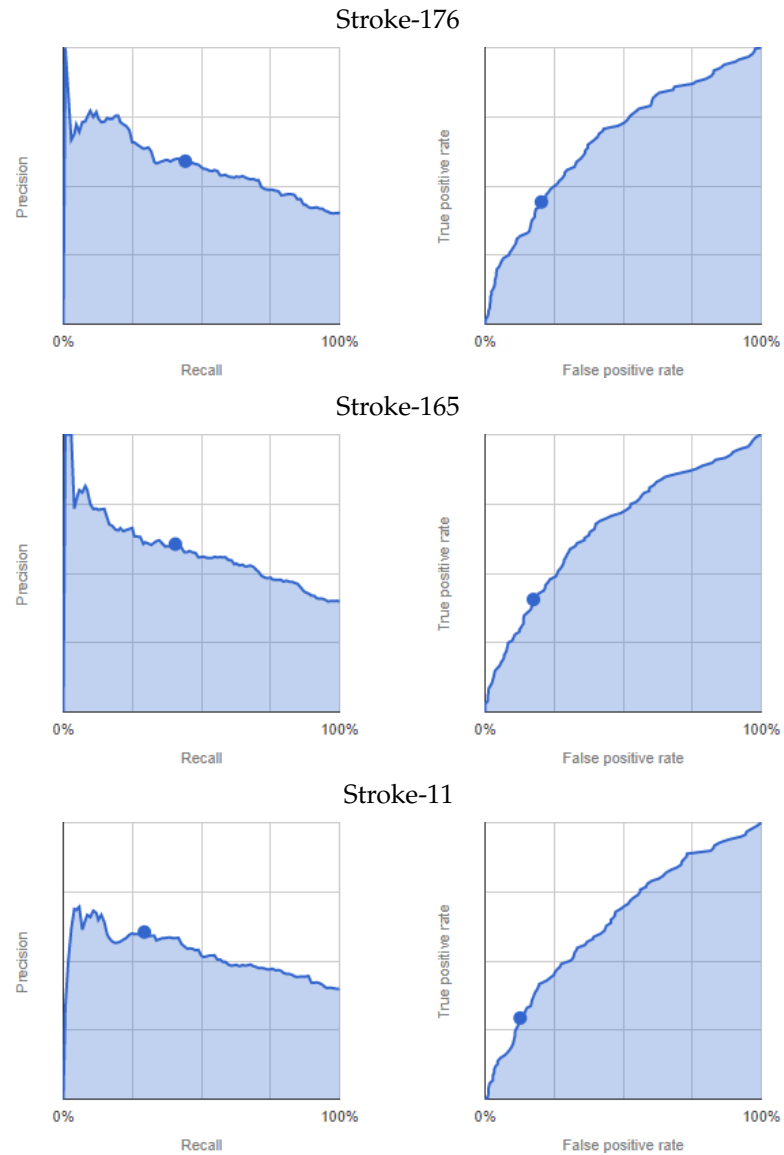


Figure 5.10: AutoML-based classification of cognitive test performance per feature subsets. Precision-recall curves (left) and ROC curves (right), dots indicate the default 0.5 score threshold set by the framework.

Feature subset	Accuracy	F1 score	Log loss	Precision	Recall	AUC (PR)	AUC (ROC)
Stroke-176	65.5%	0.735	0.630	68.3%	79.6%	0.581	0.671
Stroke-165	65.8%	0.486	0.631	60.5%	40.6%	0.582	0.668
Stroke-11	64.2%	0.395	0.646	60.4%	29.3%	0.527	0.641

Table 5.5: Results of the AutoML experiments for cognitive test performance classification with different feature subsets used for training. Data split: 80% training, 10% validation and 10% testing. Only stroke-based features are considered, because the AutoML approach requires at least 1,000 rows of data.

5.4.3 Discussion

Interestingly, the AdaBoost approach works best on this classification problem, whereas its results on the sketch recognition problems in previous experiments were rather unremarkable. It is suspected that this is due to two circumstances. First of all, the here considered problem is a binary classification problem instead of the multi-class classification in the previous experiments. In addition, a simple fine-tuning of the AdaBoost hyperparameters drastically improved the prediction accuracy. An accuracy of 87.5% with similar precision and recall values is not perfect, but it outperforms all previous approaches on the cognitive tests considered [21, 24, 91, 107].

It seems that the 11 additional features for cognitive assessments are not providing any added value. The best results are achieved on the here presented set of 165 digital pen features. An interesting observation is, that the SVMs and the Gradient Boosted Tree models seem to have found an optimum on their own for both Sketch-176 and Sketch-165, as the measurement results are the same in all six cases, only the ROC curves differ slightly. Figure 5.11 shows the top 10 averaged feature weights for the linear SVMs. All ten most important features are similar between both feature sets, only their weights differ. This might indicate that the classifiers do select more relevant features on their own during training and that providing the highest possible amount of features can be a good choice. Further investigation into this direction might be necessary to confirm or dismiss this hypothesis. Judging from the importance of features in figure 5.11, the HBF49 feature *compactness* and Willems & Niels feature *average velocity* are among the most important decision factors. Compactness models how close the sample points are to each other, which together with the average velocity leads to the assumption that cognitively impaired subjects tend to sketch noticeably slower than their healthy counterparts. This finding is supported by related work, which shows that patients with cognitive impairment experience loss of fine motor performance and that temporal measurements are higher in the cognitively impaired groups [91, 107]. However, analyzing figure 5.11 it is clear that the other top 10 features likewise have a high influence. Hence, it is not enough to look exclusively at the temporal features.

A closer look at figure 5.9 gives an intuition on the reason why the ML classifiers were able to achieve higher accuracies in previous experiments. Based on the visualization of the samples' feature vectors the reference Sign data set is separated more easily in the high dimensional space than is the case for the cognitive assessments data. Some of the healthy samples form clusters in the upper right, while the majority of points is intermixed in the middle and lower left, making it difficult for several classifiers to divide

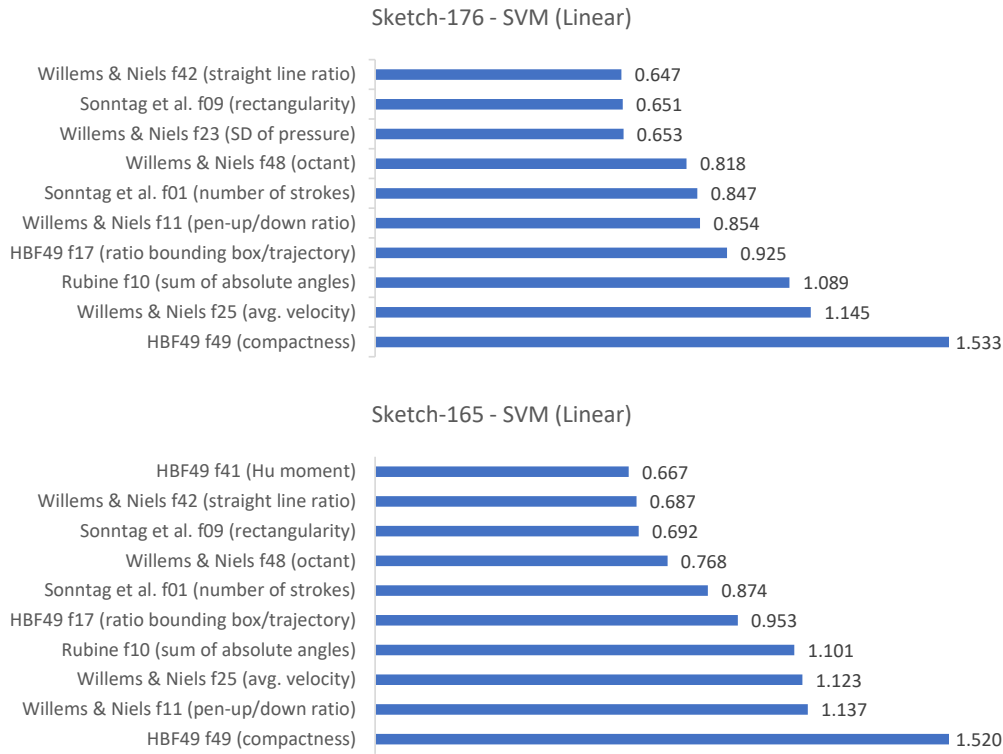


Figure 5.11: Average weights assigned to the input features of the linear SVMs (top 10).

the data appropriately. This might be an oversimplification of these complex models, but it nevertheless provides some perspective on how the models work internally.

Considering both the ten off-the-shelf ML classifiers as well as AutoML, it is clear that the stroke-level approach does not work in this setting. This could be due to several reasons. Firstly, not all of the digital pen features are necessarily well defined on single strokes, e.g., average lift duration, pen-up/pen-down ratio or straight-line ratio. Secondly, it is possible that the effects of cognitive impairment do not show equally well in all types of pen strokes, or that the features are not sensitive enough to detect this information from as little input as single strokes. Also, it might be necessary to include a more fine-grained annotation of the data. Currently, entire sketches are labeled as healthy or suspicious based on the scoring result of the entire sketch. For the stroke labeling, this sketch-wise annotation is propagated to each individual stroke. Instead, it might be more sensible to annotate individual strokes which are part of a specific error made by the subject.

Chapter 6

Conclusion

In this thesis, a state-of-the-art set of 165 digital pen features was defined, categorized, implemented and made publicly available. A comparison of this feature set against the HBF49 benchmark showed a significant improvement of sketch recognition accuracies on several data sets. Furthermore, the results of the evaluation of ten modern off-the-shelf ML classifiers showed the feature set's potential to generalize on a wide selection of data sets from different domains. In addition, the AutoML experiment proved that a proper fine-tuning and optimization of feature-based ML approaches can lead to superior recognition accuracies of up to 99% on most data sets.

Two approaches showed the utility of digital pen features for the analysis of paper-pencil-based neurocognitive assessments in the medical domain. A traditional approach showed how cognitive assessments can be analyzed as part of an interactive cognitive assessment tool using content analysis and medical scoring schemes, thereby reducing manual scoring effort and producing unbiased, explainable results. A second, innovative approach showed how cognitive test performance can be predicted by only looking at the sketch characteristics modeled by the digital pen features, without performing further semantic content analysis. Using standard ML techniques, the feature set outperformed all previous approaches on the cognitive tests considered, i.e., the Clock Drawing Test, the Rey-Osterrieth Complex Figure Test, and the Trail Making Test. It automatically scored cognitive tests with up to 87.5% accuracy in the binary classification task of categorizing sketches as healthy or suspicious. This supports more automatic, more objective and accurate diagnostics of pen sensor input, which can be used in hospitals and retirement homes to transparently evaluate cognitive performance (i.e., without explicit testing), to guide medical interventions, and to adapt cognitive training in a personalized manner.

Future work could evaluate whether this approach can be used in various intelligent user interface frameworks to evaluate and improve multimodal human-computer interaction through pen-based analytic capabilities in domains such as healthcare and education [70, 73]. For example, some approaches show how digital pen features can aid in predicting math expertise [71, 117], task difficulty and user performance [6]. This includes analyzing which features have the highest impact on the recognition results, thereby producing more transparent recognition results. Explanatory interactive machine learning (IML) [101] could be used to create interfaces which provide direct feedback to

the user and allow for the inclusion of expert knowledge into the ML process (human-in-the-loop). A prototypic example of such a system for sketch recognition was presented as part of the MIT Media Lab course on IML⁹ using the Gesture Recognition Toolkit [36].

Another promising topic that is currently being investigated is to link the digital pen features directly to the individual scoring results. Together with our colleagues at the Charité in Berlin we aim to further evaluate this aspect as part of an interactive user interface. The transparent feedback about which features contribute to which item in the scoring result helps physicians to understand why the model makes certain predictions and can therefore increase trust in the system. It might also help physicians to discover new approaches for the analysis of cognitive assessments. Furthermore, by taking into account the knowledge of domain experts as part of an IML system, trust and explainability of the underlying ML models could be further improved.

As discussed briefly in the related work section, several deep learning networks are currently emerging for various sketch related recognition tasks [16, 40, 43, 52, 62, 116]. The major challenge, to collect vast amounts of high-quality pen input required for training deep learning models, remains an open topic. Transfer learning for sketch recognition has been introduced only recently [92], but already shows a positive impact on the recognition accuracy of the How Humans Draw data set, whose classes are difficult to predict using digital pen features alone. Further investigation in that direction could improve performance for deep learning approaches on smaller data sets. Although the Google Quick Draw data set was not used in this thesis, because it only contains low-resolution finger input, it shows a viable approach for using gamification as the means to collect large sketch data sets via crowd-sourcing. Another interesting approach worth mentioning is active learning, which can be used to reduce the amount of manual annotation required to achieve a certain sketch recognition accuracy [114].

⁹<https://vimeo.com/76839534>

Bibliography

- [1] Emre Aksan, Fabrizio Pece, and Otmar Hilliges. 2018. DeepWriting: Making Digital Ink Editable via Deep Generative Modeling. *CoRR* abs/1801.08379 (2018). <http://arxiv.org/abs/1801.08379>
- [2] Çağla Çığ and Tevfik Metin Sezgin. 2015. Gaze-based prediction of pen-based virtual interaction tasks. *International Journal of Human-Computer Studies* 73 (2015), 91–106. DOI:<http://dx.doi.org/10.1016/j.ijhcs.2014.09.005>
- [3] Folami T. Alamudun, Tracy Hammond, Hong-Jun Yoon, and Georgia D. Tourassi. 2017. Geometry and Gesture-Based Features from Saccadic Eye-Movement as a Biometric in Radiology. In *Augmented Cognition. Neurocognition and Machine Learning*, Dylan D. Schmorow and Cali M. Fidopiastis (Eds.). Springer International Publishing, Cham, 123–138. DOI:http://dx.doi.org/10.1007/978-3-319-58628-1_11
- [4] Abdullah Almaksour, Eric Anquetil, Solen Quiniou, and Mohamed Cheriet. 2010. Personalizable Pen-Based Interface Using Lifelong Learning. In *2010 12th International Conference on Frontiers in Handwriting Recognition*. 188–193. DOI:<http://dx.doi.org/10.1109/ICFHR.2010.37>
- [5] Jon Almaz'n, Alicia Fornes, and Ernest Valveny. 2011. A Non-rigid Feature Extraction Method for Shape Recognition. In *2011 International Conference on Document Analysis and Recognition*. 987–991. DOI:<http://dx.doi.org/10.1109/ICDAR.2011.200>
- [6] Michael Barz, Kristin Altmeyer, Sarah Malone, Luisa Lauer, and Daniel Sonntag. 2020. Digital Pen Features Predict Task Difficulty and User Performance of Cognitive Tests. In *Proceedings of the 28th ACM Conference on User Modeling, Adaptation and Personalization (UMAP '20)*. Association for Computing Machinery, New York, NY, USA, 23–32. DOI:<http://dx.doi.org/10.1145/3340631.3394839>
- [7] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. 2006. Greedy Layer-wise Training of Deep Networks. In *Proceedings of the 19th International Conference on Neural Information Processing Systems (NIPS'06)*. MIT Press, Cambridge, MA, USA, 153–160.
- [8] James Bergstra, Daniel Yamins, and David Cox. 2013. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *International conference on machine learning*. 115–123.
- [9] Christopher M. Bishop. 2006. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg.

- [10] Suraj Bodapati, Sneha Reddy, and Sugamya Katta. 2020. Realistic Handwriting Generation Using Recurrent Neural Networks and Long Short-Term Networks. In *Proceedings of the Third International Conference on Computational Intelligence and Informatics*, K. Srujan Raju, A. Govardhan, B. Padmaja Rani, R. Sridevi, and M. Ramakrishna Murty (Eds.). Springer Singapore, Singapore, 651–661.
- [11] Christopher R. Bowie and Philip D. Harvey. 2006. Administration and interpretation of the Trail Making Test. *Nat Protoc* 1, 5 (2006), 2277–2281. DOI:<http://dx.doi.org/10.1038/nprot.2006.390>
- [12] Leo Breiman. 1996. *Bias, Variance, and Arcing Classifiers*. Technical Report. University of California, Berkeley.
- [13] Axel Brink, Ralph Niels, Roeland van Batenburg, Elisa C. van den Heuvel, and Lambert R. B. Schomaker. 2011. Towards robust writer verification by correcting unnatural slant. *Pattern Recognition Letters* 32, 3 (2011), 449 – 457. DOI:<http://dx.doi.org/10.1016/j.patrec.2010.10.010>
- [14] William Buxton. 1986. Chunking and phrasing and the design of human-computer dialogues. In *Information Processing 86*. Elsevier Science Publishers B.V.
- [15] Richard O. Canham, Stephen L. Smith, and Andy M. Tyrrell. 2000. Automated scoring of a neuropsychological test: the Rey Osterrieth complex figure. In *Proceedings of the 26th Euromicro Conference. EUROMICRO 2000. Informatics: Inventing the Future*, Vol. 2. 406–413 vol.2. DOI:<http://dx.doi.org/10.1109/EURMIC.2000.874519>
- [16] Victor Carbune, Pedro Gonnet, Thomas Deselaers, Henry Rowley, Alexander Daryin, Marcos Calvo, Li-Lun Wang, Daniel Keysers, Sandro Feuz, and Philippe Gervais. 2020. Fast Multi-language LSTM-based Online Handwriting Recognition. *International Journal on Document Analysis and Recognition (IJDAR)* (2020). DOI:<http://dx.doi.org/10.1007/s10032-020-00350-4>
- [17] Rich Caruana and Alexandru Niculescu-Mizil. 2006. An Empirical Comparison of Supervised Learning Algorithms. In *Proceedings of the 23rd International Conference on Machine Learning (ICML '06)*. Association for Computing Machinery, New York, NY, USA, 161–168. DOI:<http://dx.doi.org/10.1145/1143844.1143865>
- [18] Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A Library for Support Vector Machines. *ACM Trans. Intell. Syst. Technol.* 2, 3, Article 27 (May 2011), 27 pages. DOI:<http://dx.doi.org/10.1145/1961189.1961199>
- [19] Mauricio Cirelli and Ricardo Nakamura. 2014. A Survey on Multi-Touch Gesture Recognition and Multi-Touch Frameworks. In *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces (ITS '14)*. Association for Computing Machinery, New York, NY, USA, 35–44. DOI:<http://dx.doi.org/10.1145/2669485.2669509>
- [20] Daniel R. Coates, Johan Wagemans, and Bilge Sayim. 2017. Diagnosing the Periphery: Using the Rey-Osterrieth Complex Figure Drawing Test to Characterize Peripheral Visual Function. In *i-Perception*. DOI:<http://dx.doi.org/10.1177/2041669517705447>

- [21] Jamie Cohen, Dana Penney, Randall Davis, David Libon, Rodney Swenson, Olusola Ajilore, Anand Kumar, and Melissa Lamar. 2014. Digital clock drawing: Differentiating “thinking” versus “doing” in younger and older adults with depression. *Journal of the International Neuropsychological Society* 20, 9 (2014), 920–928. DOI: <http://dx.doi.org/10.1017/S1355617714000757>
- [22] Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine Learning* 20, 3 (1995), 273–297. DOI: <http://dx.doi.org/10.1007/BF00994018>
- [23] Jessamyn Dahmen, Diane Cook, Robert Fellows, and Maureen Schmitter-Edgecombe. 2017. An analysis of a digital variant of the Trail Making Test using machine learning techniques. *Technol Health Care* 25, 2 (2017), 251–264. DOI: <http://dx.doi.org/10.3233/THC-161274>
- [24] Randall Davis, David Libon, Roda Au, David Pitman, and Dana Penney. 2015. THink: Inferring Cognitive Status from Subtle Behaviors. *AI Magazine* 36, 3 (Sep. 2015), 49–60. DOI: <http://dx.doi.org/10.1609/aimag.v36i3.2602>
- [25] Adrien Delaye and Eric Anquetil. 2013. HBF49 feature set: A first unified baseline for online symbol recognition. *Pattern Recognition* 46, 1 (Jan. 2013), 117–130. DOI: <http://dx.doi.org/10.1016/j.patcog.2012.07.015>
- [26] Li Deng and Dong Yu. 2014. *Deep Learning: Methods and Applications*. Technical Report MSR-TR-2014-21. <https://www.microsoft.com/en-us/research/publication/deep-learning-methods-and-applications/>
- [27] Peter Drotár, Jiří Mekyska, Irena Rektorová, Lucia Masarová, Zdenek Smékal, and Marcos Faundez-Zanuy. 2014. Analysis of in-air movement in handwriting: A novel marker for Parkinson’s disease. *Computer Methods and Programs in Biomedicine* 117, 3 (2014), 405–411. DOI: <http://dx.doi.org/10.1016/j.cmpb.2014.08.007>
- [28] James F. Duley, Jean W. Wilkins, Sherry L. Hamby, Donald G. Hopkins, Rebecca D. Burwell, and Neil S. Barry. 1993. Explicit scoring criteria for the Rey-Osterrieth and Taylor complex figures. *Clinical Neuropsychologist* 7, 1 (1993), 29–38. DOI: <http://dx.doi.org/10.1080/13854049308401885>
- [29] Mathias Eitz, James Hays, and Marc Alexa. 2012. How Do Humans Sketch Objects? *ACM Trans. Graph. (Proc. SIGGRAPH)* 31, 4 (2012), 44:1–44:10.
- [30] Matthias Feurer and Frank Hutter. 2019. *Hyperparameter Optimization*. Springer International Publishing, Cham, 3–33. DOI: http://dx.doi.org/10.1007/978-3-030-05318-5_1
- [31] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. 2015. Efficient and Robust Automated Machine Learning. In *Advances in Neural Information Processing Systems* 28. Curran Associates, Inc., 2962–2970. <http://papers.nips.cc/paper/5872-efficient-and-robust-automated-machine-learning.pdf>
- [32] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Tobias Springenberg, Manuel Blum, and Frank Hutter. 2019. *Auto-sklearn: Efficient and Robust Automated Machine Learning*. Springer International Publishing, Cham, 113–134. DOI: http://dx.doi.org/10.1007/978-3-030-05318-5_6

- [33] Marshal Folstein, Susan Folstein, and Paul McHugh. 1975. "Mini-mental state". A practical method for grading the cognitive state of patients for the clinician. *Journal of Psychiatric Research* 12, 3 (Nov. 1975), 189–198. DOI:[http://dx.doi.org/10.1016/0022-3956\(75\)90026-6](http://dx.doi.org/10.1016/0022-3956(75)90026-6)
- [34] Morris Freedman, Larry Leach, Edith Kaplan, Gord Winocur, Kenneth Shulman, and Dean C. Delis. 1994. *Clock Drawing: A Neuropsychological Analysis*. Oxford University Press.
- [35] Gerald Gatterer, Peter Fischer, Margarethe Simanyi, and Walter Danielczyk. 1989. The A-K-T ("Alters-Konzentrations-Test") a new psychometric test for geriatric patients. *Funct. Neurol.* 4, 3 (1989), 273–276.
- [36] Nicholas Gillian and Joseph A. Paradiso. 2014. The Gesture Recognition Toolkit. *Journal of Machine Learning Research* 15, 101 (2014), 3483–3487. <http://jmlr.org/papers/v15/gillian14a.html>
- [37] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. <http://www.deeplearningbook.org>.
- [38] Ronald L. Graham. 1972. An Efficient Algorithm for Determining the Convex Hull of a Finite Planar Set. *Inf. Process. Lett.* 1, 4 (1972), 132–133.
- [39] Alex Graves. 2013. Generating Sequences With Recurrent Neural Networks. *CoRR* abs/1308.0850 (2013). <http://arxiv.org/abs/1308.0850>
- [40] David Ha and Douglas Eck. 2017. A Neural Representation of Sketch Drawings. *CoRR* abs/1704.03477 (2017). <http://arxiv.org/abs/1704.03477>
- [41] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA Data Mining Software: An Update. *SIGKDD Explor. Newsl.* 11, 1 (Nov. 2009), 10–18. DOI:<http://dx.doi.org/10.1145/1656274.1656278>
- [42] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2009. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer. DOI:<http://dx.doi.org/10.1007/978-0-387-84858-7>
- [43] Jun-Yan He, Xiao Wu, Yu-Gang Jiang, Bo Zhao, and Qiang Peng. 2017. Sketch Recognition with Deep Visual-Sequential Fusion Model. In *Proceedings of the 25th ACM International Conference on Multimedia (MM '17)*. Association for Computing Machinery, New York, NY, USA, 448–456. DOI:<http://dx.doi.org/10.1145/3123266.3123321>
- [44] Anika Heimann-Steinert, Antje Latendorf, Alexander Prange, Daniel Sonntag, and Ursula Müller-Werdan. 2020. Digital pen technology for conducting cognitive assessments: a cross-over study with older adults. *Psychological Research* 2020, 12 (December 2020), 1–9. DOI:<http://dx.doi.org/10.1007/s00426-020-01452-8>
- [45] Geoffrey Hinton and Ruslan Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *Science* 313, 5786 (July 2006), 504–507. DOI:<http://dx.doi.org/10.1126/science.1127647>
- [46] Heloise Hse and A. Richard Newton. 2004. Sketched symbol recognition using Zernike moments. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, Vol. 1. 367–370 Vol.1. DOI:<http://dx.doi.org/10.1109/ICPR.2004.1334128>

- [47] Ming-Kuei Hu. 1962. Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory* 8, 2 (Feb. 1962), 179–187. DOI:<http://dx.doi.org/10.1109/TIT.1962.1057692>
- [48] Laurent Hyafil and Ronald L. Rivest. 1976. Constructing optimal binary decision trees is NP-complete. *Inform. Process. Lett.* 5, 1 (1976), 15 – 17. DOI:[http://dx.doi.org/https://doi.org/10.1016/0020-0190\(76\)90095-8](http://dx.doi.org/https://doi.org/10.1016/0020-0190(76)90095-8)
- [49] Sebastiano Impedovo, Giuseppe Pirlo, Raffaele Modugno, and Anna Ferrante. 2010. Zoning Methods for Hand-Written Character Recognition: An Overview. In *2010 12th International Conference on Frontiers in Handwriting Recognition*. 329–334. DOI: <http://dx.doi.org/10.1109/ICFHR.2010.57>
- [50] IntuiDoc. 2011. IMISketchSBD Database. <https://www-intuidoc.irisa.fr/en/base-de-donnees-imisketchsdb/>. (Nov 2011). [Online; accessed 09-November-2020].
- [51] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2014. *An Introduction to Statistical Learning: With Applications in R*. Springer Publishing Company, Incorporated.
- [52] Abdullah Talha Kabakus. 2020. A Novel Sketch Recognition Model based on Convolutional Neural Networks. In *2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*. 1–6. DOI:<http://dx.doi.org/10.1109/HORA49412.2020.9152911>
- [53] Elke Kalbe, Josef Kessler, Pasquale Calabrese, R. Smith, Anthony Passmore, Matthias Brand, and Roger Bullock. 2004. DemTect: a new, sensitive cognitive screening test to support the diagnosis of mild cognitive impairment and early dementia. *International Journal of Geriatric Psychiatry* 19, 2 (Feb. 2004), 136–143. DOI:<http://dx.doi.org/10.1002/gps.1042>
- [54] Steve Klabnik and Carol Nichols. 2018. *The Rust Programming Language*. No Starch Press, USA.
- [55] Elina Kuosmanen, Valerii Kan, Aku Visuri, Simo Hosio, and Denzil Ferreira. 2020. Let’s Draw: Detecting and Measuring Parkinson’s Disease on Smartphones. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI ’20)*. Association for Computing Machinery, New York, NY, USA, 1–9. DOI:<http://dx.doi.org/10.1145/3313831.3376864>
- [56] Joseph J. LaViola and Robert C. Zeleznik. 2007. A Practical Approach for Writer-Dependent Symbol Recognition Using a Writer-Independent Symbol Recognizer. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29, 11 (nov 2007), 1917–1926. DOI:<http://dx.doi.org/10.1109/TPAMI.2007.1109>
- [57] Muriel Deutsch Lezak, Diane B. Howieson, David W. Loring, H. Julia Hannay, and Jill S. Fischer. 2004. *Neuropsychological Assessment*. Oxford University Press.
- [58] Chuang Li, Xing Zhang, Feng Lin, Zhiyong Wang, Jun’E Liu, Rui Zhang, and Haiqiang Wang. 2019. A Stroke-Based RNN for Writer-Independent Online Signature Verification. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE Computer Society, Los Alamitos, CA, USA, 526–532. DOI: <http://dx.doi.org/10.1109/ICDAR.2019.00090>

- [59] Gábor Melis, Chris Dyer, and Phil Blunsom. 2018. On the State of the Art of Evaluation in Neural Language Models. In *International Conference on Learning Representations*. <https://openreview.net/forum?id=ByJHuTgA->
- [60] Mario F. Mendez, Thomas Ala, and Kara L. Underwood. 1992. Development of Scoring Criteria for the Clock Drawing Task in Alzheimer’s Disease. *Journal of the American Geriatrics Society* 40, 11 (1992), 1095–1099. DOI:<http://dx.doi.org/10.1111/j.1532-5415.1992.tb01796.x>
- [61] Ingo Mierswa, Michael Wurst, Ralf Klinkenberg, Martin Scholz, and Timm Euler. 2006. YALE: Rapid Prototyping for Complex Data Mining Tasks. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD ’06)*. Association for Computing Machinery, New York, NY, USA, 935–940. DOI:<http://dx.doi.org/10.1145/1150402.1150531>
- [62] Momina Moetesum, Imran Siddiqi, Shoaib Ehsan, and Nicole Vincent. 2020. Deformation modeling and classification using deep convolutional neural networks for computerized analysis of neuropsychological drawings. *Neural Computing and Applications* 32, 16 (2020), 12909–12933. DOI:<http://dx.doi.org/10.1007/s00521-020-04735-8>
- [63] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. 2012. *Foundations of Machine Learning*. The MIT Press.
- [64] John C. Morris, Richard C. Mohs, Helen Rogers, Gerda G. Fillenbaum, and Albert Heyman. 1988. Consortium to establish a registry for Alzheimer’s disease (CERAD) clinical and neuropsychological assessment of Alzheimer’s disease. *Psychopharmacol Bull.* 24, 4 (1988), 641–52. DOI:<http://dx.doi.org/10.1212/WNL.39.9.1159>
- [65] Ziad S. Nasreddine, Natalie A. Phillips, Valérie Bédirian, Simon Charbonneau, Victor Whitehead, Isabelle Collin, Jeffrey L. Cummings, and Howard Chertkow. 2005. The Montreal Cognitive Assessment, MoCA: a brief screening tool for mild cognitive impairment. *Journal of the American Geriatrics Society* 53, 4 (2005), 695–699. DOI:<http://dx.doi.org/10.1111/j.1532-5415.2005.53221.x>
- [66] Franz Neyer, Juliane Felber, and Claudia Gebhardt. 2012. Entwicklung und Validierung einer Kurzskaala zur Erfassung von Technikbereitschaft. *Diagnostica* 58 (04 2012), 87–99. DOI:<http://dx.doi.org/10.1026/0012-1924/a000067>
- [67] Ralph Niels, Louis Vuurpijl, and Lambert R. B. Schomaker. 2007. Automatic Allograph Matching in Forensic Writer Identification. *International Journal of Pattern Recognition and Artificial Intelligence* 21, 01 (2007), 61–81. DOI:<http://dx.doi.org/10.1142/S0218001407005302>
- [68] Ralph Niels, Don Willems, and Louis Vuurpijl. 2008. The NicIcon Database of Handwritten Icons for Crisis Management. *Proceedings of the International Conference on Frontiers in Handwriting Recognition* (2008), 296–301. <http://unipen.nici.ru.nl/nicicon/data/publications/NicIcon.pdf>
- [69] Mira Niemann, Alexander Prange, and Daniel Sonntag. 2018. Towards a Multi-modal Multisensory Cognitive Assessment Framework. In *31st IEEE International Symposium on Computer-Based Medical Systems, CBMS 2018, Karlstad, Sweden, June 18-21, 2018*. 24–29. DOI:<http://dx.doi.org/10.1109/CBMS.2018.00012>

- [70] Sharon Oviatt. 2013. *The Design of Future Educational Interfaces*. Routledge. DOI: <http://dx.doi.org/10.4324/9780203366202>
- [71] Sharon Oviatt and Adrienne Cohen. 2014. Written Activity, Representations and Fluency as Predictors of Domain Expertise in Mathematics. In *Proceedings of the 16th International Conference on Multimodal Interaction (ICMI '14)*. Association for Computing Machinery, New York, NY, USA, 10–17. DOI:<http://dx.doi.org/10.1145/2663204.2663245>
- [72] Sharon Oviatt, Kevin Hang, Jianlong Zhou, Kun Yu, and Fang Chen. 2018. Dynamic Handwriting Signal Features Predict Domain Expertise. *ACM Trans. Interact. Intell. Syst.* 8, 3, Article 18 (July 2018), 21 pages. DOI:<http://dx.doi.org/10.1145/3213309>
- [73] Sharon Oviatt, Björn Schuller, Philip R Cohen, Daniel Sonntag, Gerasimos Potamianos, and Antonio Krüger. 2017. Introduction: Scope, Trends, and Paradigm Shift in the Field of Computer Interfaces. In *The Handbook of Multimodal-Multisensor Interfaces*. Association for Computing Machinery and Morgan & Claypool, New York, NY, USA, 1–15. DOI:<http://dx.doi.org/10.1145/3015783.3015784>
- [74] Delnaz Palsetia, G. Prasad Rao, Sarvada. C. Tiwari, Pragya Lodha, and Avinash De Sousa. 2018. The Clock Drawing Test versus Mini-mental Status Examination as a Screening Tool for Dementia: A Clinical Comparison. *Indian J Psychol Med* 40, 1 (2018), 1–10. DOI:http://dx.doi.org/10.4103/IJPSYM.IJPSYM_244_17
- [75] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-Learn: Machine Learning in Python. *J. Mach. Learn. Res.* 12, null (Nov. 2011), 2825–2830.
- [76] Alexander Prange, Michael Barz, and Daniel Sonntag. 2018. A categorisation and implementation of digital pen features for behaviour characterisation. *CoRR* abs/1810.03970 (2018). <http://arxiv.org/abs/1810.03970>
- [77] Alexander Prange, Mira Niemann, Antje Latendorf, Anika Steinert, and Daniel Sonntag. 2019. Multimodal Speech-based Dialogue for the Mini-Mental State Examination. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems, CHI 2019, Glasgow, Scotland, UK, May 04-09, 2019*. ACM. DOI: <http://dx.doi.org/10.1145/3290607.3299040>
- [78] Alexander Prange, Indra Praveen Sandrala, Markus Weber, and Daniel Sonntag. 2015. Robot Companions and Smartpens for Improved Social Communication of Dementia Patients. In *Proceedings of the 20th International Conference on Intelligent User Interfaces Companion, IUI 2015*. ACM, 65–68. DOI:<http://dx.doi.org/10.1145/2732158.2732174>
- [79] Alexander Prange and Daniel Sonntag. 2019. Modeling Cognitive Status through Automatic Scoring of a Digital Version of the Clock Drawing Test. In *Proceedings of the 27th ACM Conference on User Modeling, Adaptation and Personalization, UMAP 2019, Larnaca, Cyprus, June 9-12, 2019*. ACM, 70–77. DOI:<http://dx.doi.org/10.1145/3320435.3320452>

- [80] Yonggang Qi and Zheng-Hua Tan. 2019. SketchSegNet+: An End-to-End Learning of RNN for Multi-Class Sketch Semantic Segmentation. *IEEE Access* 7 (2019), 102717–102726. DOI:<http://dx.doi.org/10.1109/ACCESS.2019.2929804>
- [81] Marc Aurelio Ranzato, Christopher Poultney, Sumit Chopra, and Yann LeCun. 2007. Efficient learning of sparse representations with an energy-based model. In *Advances in Neural Information Processing Systems 19 - Proceedings of the 2006 Conference*. 1137–1144.
- [82] Ralph M. Reitan. 1971. Trail Making Test Results for Normal and Brain-Damaged Children. *Perceptual and Motor Skills* 33, 2 (1971), 575–581. DOI:<http://dx.doi.org/10.2466/pms.1971.33.2.575>
- [83] Ralph M. Reitan. 1986. *Trail Making Test: Manual for administration and scoring*. Reitan Neuropsychology Laboratory.
- [84] Ralph M. Reitan. 1992. *Trail Making Test*. Reitan Neuropsychology Laboratory.
- [85] Yosra Rekik, Radu-Daniel Vatavu, and Laurent Grisoni. 2014. Understanding Users’ Perceived Difficulty of Multi-Touch Gesture Articulation. In *Proceedings of the 16th International Conference on Multimodal Interaction (ICMI ’14)*. Association for Computing Machinery, New York, NY, USA, 232–239. DOI:<http://dx.doi.org/10.1145/2663204.2663273>
- [86] Ney Renau-Ferrer, Peiyu Li, Adrien Delaye, and Eric Anquetil. 2012. The ILGDB database of realistic pen-based gestural commands. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*. 3741–3744.
- [87] Joan Mas Romeu, Bart Lamiroy, Gemma Sanchez, and Josep Lladós. 2006. Automatic Adjacency Grammar Generation from User Drawn Sketches. In *18th International Conference on Pattern Recognition (ICPR’06)*, Vol. 2. 1026–1029. DOI:<http://dx.doi.org/10.1109/ICPR.2006.293>
- [88] Dean Rubine. 1991. Specifying Gestures by Example. In *Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH ’91)*. ACM, New York, NY, USA, 329–337. DOI:<http://dx.doi.org/10.1145/122718.122753>
- [89] Stuart Russell and Peter Norvig. 2009. *Artificial Intelligence: A Modern Approach* (3rd ed.). Prentice Hall Press, USA.
- [90] Timothy A. Salthouse. 2011. What cognitive abilities are involved in trail-making performance? *Intelligence* 39, 4 (July 2011), 222–232. DOI:<http://dx.doi.org/10.1016/j.intell.2011.03.001>
- [91] Andreas Schröter, Roland Mergl, Katharina Bürger, Harald Hampel, Hans-Jürgen Möller, and Ulrich Hegerl. 2003. Kinematic analysis of handwriting movements in patients with Alzheimer’s disease, mild cognitive impairment, depression and healthy subjects. *Dementia and Geriatric Cognitive Disorders* 15, 3 (2003), 132–142. DOI:<http://dx.doi.org/10.1159/000068484>
- [92] Mustafa Sert and Emel Boyacı. 2019. Sketch Recognition Using Transfer Learning. *Multimedia Tools Appl.* 78, 12 (June 2019), 17095–17112. DOI:<http://dx.doi.org/10.1007/s11042-018-7067-1>

- [93] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. 2012. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*. 2951–2959.
- [94] Daniel Sonntag. 2017. Interakt - A Multimodal Multisensory Interactive Cognitive Assessment Tool. *CoRR* abs/1709.01796 (2017). <http://arxiv.org/abs/1709.01796>
- [95] Daniel Sonntag. 2018. Interactive Cognitive Assessment Tools: A Case Study on Digital Pens for the Clinical Assessment of Dementia. *CoRR* abs/1810.04943 (2018). <http://arxiv.org/abs/1810.04943>
- [96] Daniel Sonntag, Markus Weber, Alexander Cavallaro, and Matthias Hammon. 2014. Integrating Digital Pens in Breast Imaging for Instant Knowledge Acquisition. *AI Magazine* 35, 1 (2014), 26–37. DOI:<http://dx.doi.org/10.1609/aimag.v35i1.2501>
- [97] William Souillard-Mandar, Randall Davis, Cynthia Rudin, Rhoda Au, David J. Libon, Rodney Swenson, Catherine C. Price, Melissa Lamar, and Dana L. Penney. 2016. Learning classification models of cognitive conditions from subtle behaviors in the digital Clock Drawing Test. *Machine Learning* 102, 3 (2016), 393–441. DOI:<http://dx.doi.org/10.1007/s10994-015-5529-5>
- [98] Gem Stapleton, Beryl Plimmer, Aidan Delaney, and Peter Rodgers. 2015. Combining Sketching and Traditional Diagram Editing Tools. *ACM Trans. Intell. Syst. Technol.* 6, 1, Article 10 (March 2015), 29 pages. DOI:<http://dx.doi.org/10.1145/2631925>
- [99] Ching Y. Suen, Marc Berthod, and Shunji Mori. 1980. Automatic recognition of handprinted characters—the state of the art. *Proc. IEEE* 68, 4 (1980), 469–487. DOI:<http://dx.doi.org/10.1109/PROC.1980.11675>
- [100] Kashvi Taunk, Sanjukta De, Srishti Verma, and Aleena Swetapadma. 2019. A Brief Review of Nearest Neighbor Algorithm for Learning and Classification. In *2019 International Conference on Intelligent Computing and Control Systems (ICCS)*. 1255–1260. DOI:<http://dx.doi.org/10.1109/ICCS45141.2019.9065747>
- [101] Stefano Teso and Kristian Kersting. 2019. Explanatory Interactive Machine Learning. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society (AIES '19)*. Association for Computing Machinery, New York, NY, USA, 239–245. DOI:<http://dx.doi.org/10.1145/3306618.3314293>
- [102] Tom N. Tombaugh. 2004. Trail Making Test A and B: Normative data stratified by age and education. *Archives of Clinical Neuropsychology* 19, 2 (03 2004), 203–214. DOI:[http://dx.doi.org/10.1016/S0887-6177\(03\)00039-8](http://dx.doi.org/10.1016/S0887-6177(03)00039-8)
- [103] Laurens Van Der Maaten. 2014. Accelerating t-SNE Using Tree-Based Algorithms. *J. Mach. Learn. Res.* 15, 1 (Jan. 2014), 3221–3245.
- [104] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research* 9, 86 (2008), 2579–2605. <http://jmlr.org/papers/v9/vandermaaten08a.html>

- [105] Christian Viard-Gaudin, Pierre Michel Lallican, Stefan Knerr, and Philippe Binter. 1999. The IRESTE On/Off (IRONOFF) dual handwriting database. In *Proceedings of the Fifth International Conference on Document Analysis and Recognition. ICDAR '99 (Cat. No.PR00318)*. 455–458. DOI:<http://dx.doi.org/10.1109/ICDAR.1999.791823>
- [106] Yi-Xuan Wang, Jue Zhao, Da-Ke Li, Fang Peng, Ying Wang, Ke Yang, Zhen-Yang Liu, Feng-Tao Liu, Jian-Jun Wu, and Jian Wang. 2017. Associations between cognitive impairment and motor dysfunction in Parkinson’s disease. *Brain and Behavior* 7, 6 (2017), e00719. DOI:<http://dx.doi.org/10.1002/brb3.719>
- [107] Perla Werner, Sara Rosenblum, Gady Bar-On, Jeremia Heinik, and Amos Korczyn. 2006. Handwriting process variables discriminating mild Alzheimer’s disease and mild cognitive impairment. *The Journals of Gerontology Series B: Psychological Sciences and Social Sciences* 61, 4 (2006), P228–P236. DOI:<http://dx.doi.org/10.1093/geronb/61.4.p228>
- [108] Don Willems and Ralph Niels. 2008. *Definitions for Features used in Online Pen Gesture Recognition*. Technical Report. NICI, Radboud University Nijmegen. <http://unipen.nici.ru.nl/NicIcon/>
- [109] Don Willems, Ralph Niels, Marcel van Gerven, and Louis Vuurpijl. 2009. Iconic and multi-stroke gesture recognition. *Pattern Recognition* 42, 12 (2009), 3303 – 3312. DOI:<http://dx.doi.org/10.1016/j.patcog.2009.01.030>
- [110] Don Willems, Stéphane Rossignol, and Louis Vuurpijl. 2005. Mode detection in on-line pen drawing and handwriting recognition. In *Eighth International Conference on Document Analysis and Recognition (ICDAR’05)*. 31–35 Vol. 1. DOI:<http://dx.doi.org/10.1109/ICDAR.2005.160>
- [111] Don Willems and Louis Vuurpijl. 2007. A Bayesian Network Approach to Mode Detection for Interactive Maps. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, Vol. 2. 869–873. DOI:<http://dx.doi.org/10.1109/ICDAR.2007.4377039>
- [112] Blake Williford, Matthew Runyon, and Tracy Hammond. 2020. Recognizing Perspective Accuracy: An Intelligent User Interface for Assisting Novices. In *Proceedings of the 25th International Conference on Intelligent User Interfaces (IUI '20)*. Association for Computing Machinery, New York, NY, USA, 231–242. DOI:<http://dx.doi.org/10.1145/3377325.3377511>
- [113] Xindong Wu, Vipin Kumar, J Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J McLachlan, Angus Ng, Bing Liu, Philip S Yu, Zhi-Hua Zhou, Michael Steinbach, David J Hand, and Dan Steinberg. 2008. Top 10 algorithms in data mining. *Knowledge and Information Systems* 14, 1 (2008), 1–37. DOI:<http://dx.doi.org/10.1007/s10115-007-0114-2>
- [114] Erelcan Yanik and Tevfik Metin Sezgin. 2015. Active learning for sketch recognition. *Computers & Graphics* 52 (2015), 93–105. DOI:<http://dx.doi.org/https://doi.org/10.1016/j.cag.2015.07.023>
- [115] Harry Zhang. 2004. The Optimality of Naive Bayes. *Proceedings of the Seventeenth International Florida Artificial Intelligence Research Society Conference, FLAIRS 2004* 2 (01 2004), 6. <https://www.aaai.org/Papers/FLAIRS/2004/Flairs04-097.pdf>

- [116] Jianhui Zhang, Yilan Chen, Lei Li, Hongbo Fu, and Chiew-Lan Tai. 2018. Context-Based Sketch Classification. In *Proceedings of the Joint Symposium on Computational Aesthetics and Sketch-Based Interfaces and Modeling and Non-Photorealistic Animation and Rendering (Expressive '18)*. Association for Computing Machinery, New York, NY, USA, Article 3, 10 pages. DOI:<http://dx.doi.org/10.1145/3229147.3229154>
- [117] Jianlong Zhou, Kevin Hang, Sharon Oviatt, Kun Yu, and Fang Chen. 2014. Combining Empirical and Machine Learning Techniques to Predict Math Expertise Using Pen Signal Features. In *Proceedings of the 2014 ACM Workshop on Multimodal Learning Analytics Workshop and Grand Challenge (MLA '14)*. Association for Computing Machinery, New York, NY, USA, 29–36. DOI:<http://dx.doi.org/10.1145/2666633.2666638>

Appendix A

Rubine's Feature Set

Features from this section are implementations of the described features by Rubine [88].

A.1 Cosine of initial angle

$$f_1 = \frac{(x_2 - x_0)}{\sqrt{(x_2 - x_0)^2 + (y_2 - y_0)^2}} \quad (\text{A.1})$$

A.2 Sine of initial angle

$$f_2 = \frac{(y_2 - y_0)}{\sqrt{(x_2 - x_0)^2 + (y_2 - y_0)^2}} \quad (\text{A.2})$$

A.3 Length of bounding box diagonal

$$f_3 = \sqrt{(x_{\max} - x_{\min})^2 + (y_{\max} - y_{\min})^2} \quad (\text{A.3})$$

A.4 Angle of the bounding box diagonal

$$f_4 = \arctan \frac{y_{\max} - y_{\min}}{x_{\max} - x_{\min}} \quad (\text{A.4})$$

A.5 Distance between first and last point

$$f_5 = \sqrt{(x_{n-1} - x_0)^2 + (y_{n-1} - y_0)^2} \quad (\text{A.5})$$

A.6 Cosine of the angle between first and last point

$$f_6 = \frac{(x_{n-1} - x_0)}{f_5} \quad (\text{A.6})$$

A.7 Sine of the angle between first and last point

$$f_7 = \frac{(y_{n-1} - y_0)}{f_5} \quad (\text{A.7})$$

A.8 Total gesture length

Let $\Delta x_i = x_{i+1} - x_i$, $\Delta y_i = y_{i+1} - y_i$

$$f_8 = \sum_{i=0}^{n-2} \sqrt{\Delta x_i^2 + \Delta y_i^2} \quad (\text{A.8})$$

A.9 Total angle traversed

Let

$$\theta_i = \arctan \frac{\Delta x_i \Delta y_{i-1} - \Delta x_{i-1} \Delta y_i}{\Delta x_i \Delta x_{i-1} + \Delta y_i \Delta y_{i-1}} \quad (\text{A.9})$$

$$f_9 = \sum_{i=1}^{n-2} \theta_i \quad (\text{A.10})$$

A.10 Sum of the absolute value of the angle at each point

$$f_{10} = \sum_{i=1}^{n-2} |\theta_i| \quad (\text{A.11})$$

A.11 Sum of the squared value of the angle at each point

$$f_{11} = \sum_{i=1}^{n-2} \theta_i^2 \quad (\text{A.12})$$

A.12 Maximum speed (squared) of the gesture

Let $\Delta t_i = t_{i+1} - t_i$

$$f_{12} = \max_{i=0}^{n-2} \frac{\Delta x_i^2 + \Delta y_i^2}{\Delta t_i^2} \quad (\text{A.13})$$

A.13 Duration of the gesture

$$f_{13} = t_{n-1} - t_0 \quad (\text{A.14})$$

Appendix B

Willems & Niels Feature Set

The features described in this section are implementations based on the feature set described by Willems and Niels [108]. For this section we use the following notations:

Let c be center of the bounding box around the gesture defined by the co-ordinate axes.

$$c = \begin{pmatrix} x_{center} \\ y_{center} \end{pmatrix} = \begin{pmatrix} x_{min} + \frac{1}{2}(x_{max} - x_{min}) \\ y_{min} + \frac{1}{2}(y_{max} - y_{min}) \end{pmatrix} \quad (B.1)$$

While the ratio of the co-ordinate axis is not rotation independent, the ratio of the principal axes is. To determine the principal axes *Principal Component Analysis* is used [108]. Let p_1 and p_2 be the normalized principal component vectors of the set S . And let c be the center of the box enclosing the trajectory and along the principal component vectors. The lengths of the major axes along the principal component vectors are

$$\alpha = 2 \max_{0 \leq i < n} |p_2 \cdot (c - s_i)|, \quad \beta = 2 \max_{0 \leq i < n} |p_1 \cdot (c - s_i)| \quad (B.2)$$

B.1 Length of the gesture

$$f_1 = \sum_{i=0}^{n-2} \|s_{i+1} - s_i\| \quad (B.3)$$

B.2 Area

The area around the gesture is calculated using Graham's convex hull algorithm [38].

$$f_2 = A \quad (B.4)$$

B.3 Compactness

Let l be the length of the perimeter of the convex hull, then compactness is defined as

$$f_3 = \frac{l^2}{A} \quad (\text{B.5})$$

B.4 Ratio between co-ordinate axes

The lengths along the two co-ordinate axes (a , along the x-axis, and b along the y-axis) as given as

$$a = \max_{0 \leq i < j < n} |x_i - x_j| \quad (\text{B.6})$$

$$b = \max_{0 \leq i < j < n} |y_i - y_j| \quad (\text{B.7})$$

Eccentricity is a measure for the ratio between the co-ordinate axes.

$$f_4 = \sqrt{1 - \frac{b'^2}{a'^2}} \quad (\text{B.8})$$

where $a' = a \wedge b' = b$ if $a > b$ else $a' = b \wedge b' = a$.

B.5 Ratio between co-ordinate axes

The ratio of the co-ordinate axes, which is very much related to eccentricity, is denoted as follows

$$f_5 = \frac{b'}{a'} \quad (\text{B.9})$$

B.6 Closure

$$f_6 = \frac{f_{64}}{f_1} = \frac{\sum_{i=0}^{n-2} \|s_{i+1} - s_i\|}{\|s_{n-1} - s_0\|} \quad (\text{B.10})$$

B.7 Circular variance

$$f_7 = \frac{\sum_{i=0}^{n-1} (\|s_i - \mu\| - f_{68})^2}{n \cdot f_{68}^2} \quad (\text{B.11})$$

B.8 Curvature

Let angle between sequenced samples be:

$$\psi_{s_i} = \arccos \left\{ \frac{(s_i - s_{i-1}) \cdot (s_{i+1} - s_i)}{\|s_i - s_{i-1}\| \|s_{i+1} - s_i\|} \right\} \quad (\text{B.12})$$

then curvature will be:

$$f_8 = \sum_{i=1}^{n-2} \psi_{s_i} \quad (\text{B.13})$$

B.9 Average curvature

$$f_9 = \frac{1}{n-2} \sum_{i=1}^{n-2} \psi_{s_i} \quad (\text{B.14})$$

B.10 Standard deviation in curvature

$$f_{10} = \sqrt{\frac{1}{n-2} \sum_{i=1}^{n-2} (\psi_{s_i} - f_9)^2} \quad (\text{B.15})$$

B.11 Pen up/down ratio

Let $\mathcal{G} = \{S_0, S_1, \dots, S_{n-1}\}$ be the set of strokes composing a gesture of n strokes. The duration of stroke S of length m is given by

$$\phi(S) = t_{m-1}(S) - t_0(S) \quad (\text{B.16})$$

where $t_j(S)$ is the j -th timestamp of stroke S . The duration of all strokes \mathcal{S} with $|\mathcal{S}| = n$ is then defined as

$$\phi(\mathcal{S}) = \sum_{i=0}^{n-1} \phi(S_i) \quad (\text{B.17})$$

and the duration of the entire gesture is given by

$$\phi(\mathcal{G}) = t_{|S_{n-1}|-1}(S_{n-1}) - t_0(S_0) \quad (\text{B.18})$$

The ratio of pen up/down is the ratio between the time spent writing (pen down) and in air (pen up)

$$f_{11} = \frac{\text{airtime}}{\text{writetime}} = \frac{\phi(\mathcal{G}) - \phi(\mathcal{S})}{\phi(\mathcal{S})} \quad (\text{B.19})$$

B.12 Average direction

$$f_{12} = \frac{1}{n-1} \sum_{i=0}^{n-2} \arctan \frac{y_{i+1} - y_i}{x_{i+1} - x_i} \quad (\text{B.20})$$

B.13 Perpendicularity

$$f_{13} = \sum_{i=1}^{n-2} \sin^2 \psi_{s_i} \quad (\text{B.21})$$

B.14 Average perpendicularity

$$f_{14} = \frac{1}{n-2} \sum_{i=1}^{n-2} \sin^2 \psi_{s_i} \quad (\text{B.22})$$

B.15 Standard deviation in perpendicularity

$$f_{15} = \sqrt{\frac{1}{n-2} \sum_{i=1}^{n-2} (\sin^2 \psi_{s_i} - f_{14})^2} \quad (\text{B.23})$$

B.16 Centroid offset

The principal axes are used to calculate the centroid offset:

$$f_{16} = |\mathbf{p}_2 \cdot (\boldsymbol{\mu} - \mathbf{c})| \quad (\text{B.24})$$

B.17 Length of first principal axis

Based on the principal axis, its length is another feature

$$f_{17} = \alpha \quad (\text{B.25})$$

B.18 Sine orientation of principal axis

The orientation of the principal axis ψ is given by

$$f_{18} = \sin \psi = p_{1_y} \quad (\text{B.26})$$

B.19 Cosine orientation of principal axis

$$f_{19} = \cos \psi = p_{1_x} \quad (\text{B.27})$$

B.20 Rectangularity

Based on the lengths of the major axes along the principal component vectors and the area of the convex hull A , the rectangularity is defined as:

$$f_{20} = \frac{A}{\alpha \cdot \beta} \quad (\text{B.28})$$

B.21 Maximum angular difference

$$f_{21} = \max_{1+k \leq i \leq n-k} \psi_{s_i}^k \quad (\text{B.29})$$

B.22 Average pressure

$$f_{22} = \frac{1}{n} \sum_{i=0}^{n-1} p_i \quad (\text{B.30})$$

B.23 Standard deviation of pressure

$$f_{23} = \sqrt{\frac{1}{n} \sum_{i=0}^{n-1} (p_i - f_{22})^2} \quad (\text{B.31})$$

B.24 Duration

$$f_{24} = t_{n-1} - t_0 \quad (\text{B.32})$$

B.25 Average velocity

$$\mathbf{v}_i = \frac{\|s_{i+1} - s_i\| + \|s_i - s_{i-1}\|}{t_{i+1} - t_{i-1}} \quad (\text{B.33})$$

$$f_{25} = \frac{1}{n-2} \sum_{i=1}^{n-2} \|\mathbf{v}_i\| \quad (\text{B.34})$$

B.26 Standard deviation of velocity

$$f_{26} = \sqrt{\frac{1}{n-2} \sum_{i=1}^{n-2} (\|\mathbf{v}_i\| - f_{25})^2} \quad (\text{B.35})$$

B.27 Maximum velocity

$$f_{27} = \max_{1 \leq i \leq n-2} \|\mathbf{v}_i\| \quad (\text{B.36})$$

B.28 Average acceleration

$$\mathbf{a}_i = \frac{\mathbf{v}_{i+1} - \mathbf{v}_{i-1}}{t_{i+1} - t_{i-1}} \quad (\text{B.37})$$

$$f_{28} = \frac{1}{n-4} \sum_{i=2}^{n-3} \|\mathbf{a}_i\| \quad (\text{B.38})$$

B.29 Standard deviation of acceleration

$$f_{29} = \sqrt{\frac{1}{n-4} \sum_{i=2}^{n-3} (\|\mathbf{a}_i\| - f_{28})^2} \quad (\text{B.39})$$

B.30 Maximum acceleration

$$f_{30} = \max_{2 \leq i \leq n-3} \|\mathbf{a}_i\| \quad (\text{B.40})$$

B.31 Minimum acceleration

$$f_{31} = \min_{2 \leq i \leq n-3} \|\mathbf{a}_i\| \quad (\text{B.41})$$

B.32 Number of cups

This feature counts the number of cups in the gesture [108]. Using a sliding window approach with window size t_w and a threshold t_a , the number of cups is computed as follows:

```

nCups = 0;
pos = 0;
for i = 1 to N - t_w do:
    if  $\alpha_i > t_a$  and (pos == 0 or pos < i - t_w):
        pos = i;
        nCups++;
return nCups;

```

$$f_{32} = nCups \quad (\text{B.42})$$

$$\alpha_i = \arccos\left(\frac{s_i * s_i + t_w}{|s_i| * |s_i + t_w|}\right) \quad (\text{B.43})$$

t_a is the minimal angle between the first and last segment of the window:

- Let v_1 be the directional vector between s_i and s_{i+1} .
- Let v_2 be the directional vector between s_{i+t_w} and s_{i+t_w-1} .

$$t'_a = \arccos\left(\frac{v_1 * v_2}{|v_1| * |v_2|}\right) \quad (\text{B.44})$$

```

If  $t'_a > 90$ :
     $t_a = \frac{360 - (t'_a * 2)}{2}$ 
else:
     $t_a = t'_a$ 

```

B.33 Offset of the first cup

Based on f_{32} the first cup offset is computed.

$$f_{33} = firstCupOffset \quad (\text{B.45})$$

B.34 Offset of the last cup

Based on f_{32} the last cup offset is computed.

$$f_{34} = lastCupOffset \quad (\text{B.46})$$

B.35 Initial horizontal offset

$$f_{35} = \frac{x_0 - x_{\min}}{a} \quad (\text{B.47})$$

B.36 Final horizontal offset

$$f_{36} = \frac{x_{n-1} - x_{\min}}{a} \quad (\text{B.48})$$

B.37 Initial vertical offset

$$f_{37} = \frac{y_0 - y_{\min}}{b} \quad (\text{B.49})$$

B.38 Final vertical offset

$$f_{38} = \frac{y_{n-1} - y_{\min}}{b} \quad (\text{B.50})$$

B.39 Number of straight lines

Based on the definition of straight lines by Willems & Niels [108], we denote the set of straight lines L_i inside a gesture as $\mathcal{L} = \{L_0, L_1, \dots, L_{i-1}\}$ and the number of straight lines as

$$f_{39} = |\mathcal{L}| \quad (\text{B.51})$$

B.40 Average length of straight lines

Let $\|L_i\|$ be the length of a straight line, then the average length of straight lines is calculated as

$$f_{40} = \frac{1}{|\mathcal{L}|} \sum_{i=0}^{|\mathcal{L}|-1} \|L_i\| \quad (\text{B.52})$$

B.41 Standard deviation of straight line length

$$f_{41} = \sqrt{\frac{1}{|\mathcal{L}|} \sum_{i=0}^{|\mathcal{L}|-1} (\|L_i\| - f_{40})^2} \quad (\text{B.53})$$

B.42 Straight line ratio

$$f_{42} = \sum_{i=0}^{|\mathcal{L}|-1} \frac{\|L_i\|}{\sum_{j=1}^{n-1} \|s_j - s_{j-1}\|} = \frac{1}{f_1} \sum_{i=0}^{|\mathcal{L}|-1} \|L_i\| \quad (\text{B.54})$$

B.43 Largest straight line ratio

$$f_{43} = \max_{0 \leq i \leq n-1} \frac{\|L_i\|}{\sum_{j=1}^{n-1} \|s_j - s_{j-1}\|} = \frac{1}{f_1} \max_{0 \leq i < n < |\mathcal{L}|} \|L_i\| \quad (\text{B.55})$$

B.44 Number of pen down events

Let $\mathcal{G} = \{S_0, S_1, \dots, S_{i-1}\}$ be the set of strokes composing a gesture of n strokes. The number of pen down events equals the number of strokes

$$f_{44} = |\mathcal{G}| \quad (\text{B.56})$$

B.45 Octants

$$f_{44+o} = \frac{1}{n-1} \sum_{i=0}^{n-1} \omega_{io} \quad (\text{B.57})$$

where

$$\omega_{io} = \begin{cases} 1 & \text{if } \frac{\pi}{4}(o-1) \leq \nu_i < \frac{\pi}{4}o \\ 0 & \text{if } \nu_i < \frac{\pi}{4}(o-1) \vee \nu_i \geq \frac{\pi}{4}o \end{cases} \quad (\text{B.58})$$

and where

$$\nu_i = \arctan \frac{dy_i}{dx_i} \quad (\text{B.59})$$

and

$$dx_i = x_i - x_{center} \quad (\text{B.60})$$

$$dy_i = y_i - y_{center} \quad (\text{B.61})$$

B.46 Number of connecting strokes

We define the set of connected components as $\mathcal{C} = \{C_0, C_1, \dots, C_{i-1}\}$. A connected component C_i is a part of a gesture that consists of one or more strokes that touch each other, and that do not touch any other strokes [108].

$$f_{53} = |\mathcal{C}| \quad (\text{B.62})$$

B.47 Number of crossings

$$f_{54} = \sum_{i=1}^{n-2} \sum_{j=i+1}^{n-1} \kappa_{ij} \quad (\text{B.63})$$

where

$$\kappa_{ij} = \begin{cases} 1 & \text{if } s_i \rightarrow s_{i+1} \cap s_j \rightarrow s_{j+1} \neq \emptyset \\ 0 & \text{if } s_i \rightarrow s_{i+1} \cap s_j \rightarrow s_{j+1} = \emptyset \end{cases} \quad (\text{B.64})$$

B.48 Cosine of initial angle

$$f_{55} = \frac{x_2 - x_0}{\|s_2 - s_0\|} \quad (\text{B.65})$$

B.49 Sine of initial angle

$$f_{56} = \frac{y_2 - y_0}{\|s_2 - s_0\|} \quad (\text{B.66})$$

B.50 Length of the bounding box diagonal

Given the two co-ordinate axes the length of the bounding box is given as

$$f_{57} = \sqrt{a^2 + b^2} \quad (\text{B.67})$$

B.51 Angle of the bounding box diagonal

$$f_{58} = \tan \frac{b}{a} \quad (\text{B.68})$$

B.52 Length between first and last point

$$f_{59} = \|s_{n-1} - s_0\| \quad (\text{B.69})$$

B.53 Cosine of first to last point

$$f_{60} = \frac{x_{n-1} - x_0}{\|s_{n-1} - s_0\|} \quad (\text{B.70})$$

B.54 Sine of first to last point

$$f_{61} = \frac{y_{n-1} - y_0}{\|s_{n-1} - s_0\|} \quad (\text{B.71})$$

B.55 Absolute curvature

$$f_{62} = \sum_{i=1}^{n-2} |\psi_{s_i}| \quad (\text{B.72})$$

B.56 Squared curvature

$$f_{63} = \sum_{i=1}^{n-2} \psi_{s_i}^2 \quad (\text{B.73})$$

B.57 Macro perpendicularity

Let angle between sampled points be:

$$\psi_{s_i}^k = \arccos \left\{ \frac{(s_i - s_{i-k}) \cdot (s_{i+k} - s_i)}{\|s_i - s_{i-k}\| \|s_{i+k} - s_i\|} \right\} \quad (\text{B.74})$$

then macro perpendicularity will be:

$$f_{64} = \sum_{i=1+k}^{n-k} \sin^2 \psi_{s_i}^k \quad (\text{B.75})$$

B.58 Average macro perpendicularity

$$f_{65} = \frac{1}{n-2k} \sum_{i=1+k}^{n-k} \sin^2 \psi_{s_i}^k \quad (\text{B.76})$$

B.59 Standard deviation in macro perpendicularity

$$f_{66} = \sqrt{\frac{1}{n-2k} \sum_{i=1+k}^{n-k} (\sin^2 \psi_{s_i}^k - f_{65})^2} \quad (\text{B.77})$$

B.60 Ratio of principal axes

Based on the lengths of the major axes along the principal component vectors, the ratio of the principal axes becomes:

$$f_{67} = \frac{\beta}{\alpha} \quad (\text{B.78})$$

B.61 Average centroidal radius

The average distance of sample points from the centroid is a feature called average centroidal radius.

$$f_{68} = \frac{1}{n} \sum_{i=0}^{n-1} \|s_i - \mu\| \quad (\text{B.79})$$

B.62 Standard deviation of the centroidal radius

$$f_{69} = \sqrt{\frac{1}{n} \sum_{i=0}^{n-1} (\|s_i - \mu\| - f_{68})^2} \quad (\text{B.80})$$

B.63 Chain codes

Let the chain code be defined as

$$C_s = \begin{cases} 1 & \text{if } 0 \leq \psi_s < \frac{\pi}{4} \\ 2 & \text{if } \frac{\pi}{4} \leq \psi_s < \frac{\pi}{2} \\ 3 & \text{if } \frac{\pi}{2} \leq \psi_s < \frac{3\pi}{4} \\ 4 & \text{if } \frac{3\pi}{4} \leq \psi_s < \pi \\ 5 & \text{if } \pi \leq \psi_s < \frac{5\pi}{4} \\ 6 & \text{if } \frac{5\pi}{4} \leq \psi_s < \frac{3\pi}{2} \\ 7 & \text{if } \frac{3\pi}{2} \leq \psi_s < \frac{7\pi}{4} \\ 8 & \text{if } \frac{7\pi}{4} \leq \psi_s < 2\pi \end{cases} \quad (\text{B.81})$$

then the average angle of the chain code will be

$$\psi_{C_s} = \frac{(C_s - \frac{1}{2})\pi}{4} \quad (\text{B.82})$$

then

$$f_{68+2s} = \sin \psi_{C_s} \quad (\text{B.83})$$

and

$$f_{69+2s} = \cos \psi_{C_s} \quad (\text{B.84})$$

B.64 Average stroke length

If $S_i \in \mathcal{S}$ is a stroke with n sample points, then let L_i be the length of that stroke:

$$L_i = \sum_{j=0}^{n-2} \|s_{j+1} - s_j\| \quad (\text{B.85})$$

Assuming $|\mathcal{S}| = m$ the average stroke length is given by

$$f_{86} = \frac{1}{m} \sum_{i=0}^{m-1} L_i \quad (\text{B.86})$$

B.65 Standard deviation in stroke length

$$f_{87} = \sqrt{\frac{1}{m} \sum_{i=0}^{m-1} (L_i - f_{86})^2} \quad (\text{B.87})$$

B.66 Average stroke direction

If $S_i \in \mathcal{S}$ is a stroke with n sample points, then let φ_i be the direction of that stroke:

$$\varphi_i = \frac{1}{n-1} \sum_{j=0}^{n-2} \arctan \frac{y_{j+1} - y_j}{x_{j+1} - x_j} \quad (\text{B.88})$$

Assuming $|S| = m$ the average stroke direction is given by

$$f_{88} = \frac{1}{m} \sum_{i=0}^{m-1} \varphi_i \quad (\text{B.89})$$

B.67 Standard deviation in stroke direction

$$f_{89} = \sqrt{\frac{1}{m} \sum_{i=0}^{m-1} (\varphi_i - f_{88})^2} \quad (\text{B.90})$$

Appendix C

HBF49 Feature Set

The features described in this section are implementations based on the feature set HBF49 described by Delaye and Anquetil [25]. For this section we use the following notations:

Let B be the rectangular bounding box defined by $x_{min}, x_{max}, y_{min}, y_{max}$. The width w and height h of this box are defined as

$$w = x_{max} - x_{min}, h = y_{max} - y_{min} \quad (C.1)$$

Coordinates c_x and c_y are the coordinates of the center point c of bounding box B .

Let $L_{i,j}$ be the length of the path between sample points s_i and s_j , then L is the total length of the path of the gesture.

Let \mathcal{S} be the set of strokes composing the gesture.

C.1 Horizontal position of first point

Let $l = \max(h, w)$ be the side of a square box centered on c . The normalized position of the first point is then given by

$$f_1 = \frac{x_0 - c_x}{l} + \frac{1}{2} \quad (C.2)$$

C.2 Vertical position of first point

$$f_2 = \frac{y_0 - c_y}{l} + \frac{1}{2} \quad (C.3)$$

C.3 Horizontal position of last point

$$f_3 = \frac{x_{n-1} - c_x}{l} + \frac{1}{2} \quad (C.4)$$

C.4 Vertical position of last point

$$f_4 = \frac{y_{n-1} - c_y}{l} + \frac{1}{2} \quad (\text{C.5})$$

C.5 First point to last point vector length

$$v = \overrightarrow{s_0 s_{n-1}} \quad (\text{C.6})$$

$$f_5 = \|v\| \quad (\text{C.7})$$

C.6 Sine of first point to last point vector

Let u_x be the unit vector co-directional with the x axis.

$$f_6 = \frac{v_x \cdot u_x}{f_5} \quad (\text{C.8})$$

C.7 Cosine of first point to last point vector

Let u_y be the unit vector co-directional with the y axis.

$$f_7 = \frac{v_y \cdot u_y}{f_5} \quad (\text{C.9})$$

C.8 Closure

$$f_8 = \frac{\|v\|}{L}. \quad (\text{C.10})$$

C.9 Sine of initial angle

Our initial vector between the first and third point is given by $w = \overrightarrow{s_0 s_2}$.

$$f_9 = \frac{w_x \cdot u_x}{\|w\|} \quad (\text{C.11})$$

C.10 Cosine of initial angle

$$f_{10} = \frac{w_y \cdot u_y}{\|w\|} \quad (\text{C.12})$$

C.11 Horizontal inflexion

Let s_m be the middle-path point with respect to the middle point of segment s_0s_{n-1} .

$$f_{11} = \frac{1}{w} \left(x_m - \frac{x_1 + x_n}{2} \right) \quad (\text{C.13})$$

C.12 Vertical inflexion

$$f_{12} = \frac{1}{h} \left(y_m - \frac{y_1 + y_n}{2} \right) \quad (\text{C.14})$$

C.13 Downstroke proportion

Downstrokes are portions of drawing trajectories oriented towards the bottom of the writing surface, i.e. oriented towards increasing values in dimension y.

$$f_{13} = \sum_{k=0} L_k \quad \{\forall k \in \mathcal{S} \mid k \text{ is a downstroke}\} \quad (\text{C.15})$$

C.14 Number of strokes

$$f_{14} = |\mathcal{S}| \quad (\text{C.16})$$

C.15 Angle of the bounding box diagonal

$$f_{15} = \arctan \frac{h}{w} \quad (\text{C.17})$$

C.16 Trajectory length

$$f_{16} = L \quad (\text{C.18})$$

C.17 Ratio between bounding box and trajectory length

$$f_{17} = \frac{w + h}{L} \quad (\text{C.19})$$

C.18 Deviation

$$f_{18} = \frac{1}{n} \sum_{i=0}^{n-1} \|s_i \mu\| \quad (\text{C.20})$$

C.19 Average direction

$$f_{19} = \frac{1}{n-1} \sum_{i=0}^{n-2} \arctan \left(\frac{y_{i+1} - y_i}{x_{i+1} - x_i} \right) \quad (\text{C.21})$$

C.20 Curvature

$$\theta_i = \arccos \frac{\overrightarrow{s_{i-1}s_i} \cdot \overrightarrow{s_i s_{i+1}}}{\|\overrightarrow{s_{i-1}s_i}\| \|\overrightarrow{s_i s_{i+1}}\|} \quad (\text{C.22})$$

$$f_{20} = \sum_{i=1}^{n-2} \theta_i \quad (\text{C.23})$$

C.21 Perpendicularity

$$f_{21} = \sum_{i=1}^{n-2} \sin^2(\theta_i) \quad (\text{C.24})$$

C.22 k-Perpendicularity

$$\theta_i^k = \arccos \frac{\overrightarrow{s_{i-k}s_i} \cdot \overrightarrow{s_i s_{i+k}}}{\|\overrightarrow{s_{i-k}s_i}\| \|\overrightarrow{s_i s_{i+k}}\|} \quad (\text{C.25})$$

$$f_{22} = \sum_{i=k}^{n-k-1} \sin^2(\theta_i^k) \quad (\text{C.26})$$

C.23 k-Angle

$$f_{23} = \max_{i=k}^{n-k-1} \theta_i^k \quad (\text{C.27})$$

C.24 Dominant direction

Let n_a be the number of segments in \mathcal{S} ($n_a = n - K$, with K the number of strokes).

$$f_{24} = \frac{h_1 + h_5}{n_a} \quad (\text{C.28})$$

$$f_{25} = \frac{h_2 + h_6}{n_a} \quad (\text{C.29})$$

$$f_{26} = \frac{h_3 + h_7}{n_a} \quad (\text{C.30})$$

$$f_{27} = \frac{h_4 + h_8}{n_a} \quad (\text{C.31})$$

C.25 Local changes in direction

Local angle benefit from smoothing by linear combination of θ_i and θ_i^k (refer to f_{14} and f_{16}):

$$\psi_i^k = \gamma\theta_i + (1 - \gamma)\theta_i^k \quad (\text{C.32})$$

In the HBF49 feature set [25] the values are set empirically to $\gamma = 0.25$ and $k = 2$. The contributions of ψ_i^k angles are accumulated in four histogram bins uniformly distributed in $[0, \pi]$. Contributions to the histogram are weighted by the inverse of their angular distance with the central direction of the two neighboring bins. The features are obtained from the histogram h divided by n_a (see f_{24} - f_{27}).

$$f_{28} = \frac{h[0]}{n_a} \quad (\text{C.33})$$

$$f_{29} = \frac{h[1]}{n_a} \quad (\text{C.34})$$

$$f_{30} = \frac{h[2]}{n_a} \quad (\text{C.35})$$

$$f_{31} = \frac{h[3]}{n_a} \quad (\text{C.36})$$

C.26 2D histogram

For the 2D histogram we divide the rectangular bounding box around the figure into 3×3 partitions of equal size. Sampling points are sorted into the 9 cells resulting from partitioning [49]. For each sample point a fuzzy weighted contribution to the 4 neighboring cells is computed, where the weights depend on the distance from the point to the cell centers [5].

$$f_{32} = \frac{1}{n} \sum_{i=0}^{n-1} \mu_{11}(s_i) \quad (\text{C.37})$$

$$f_{33} = \frac{1}{n} \sum_{i=0}^{n-1} \mu_{12}(s_i) \quad (\text{C.38})$$

$$f_{34} = \frac{1}{n} \sum_{i=0}^{n-1} \mu_{13}(s_i) \quad (\text{C.39})$$

$$f_{35} = \frac{1}{n} \sum_{i=0}^{n-1} \mu_{21}(s_i) \quad (\text{C.40})$$

$$f_{36} = \frac{1}{n} \sum_{i=0}^{n-1} \mu_{22}(s_i) \quad (\text{C.41})$$

$$f_{37} = \frac{1}{n} \sum_{i=0}^{n-1} \mu_{23}(s_i) \quad (\text{C.42})$$

$$f_{38} = \frac{1}{n} \sum_{i=0}^{n-1} \mu_{31}(s_i) \quad (\text{C.43})$$

$$f_{39} = \frac{1}{n} \sum_{i=0}^{n-1} \mu_{32}(s_i) \quad (\text{C.44})$$

$$f_{40} = \frac{1}{n} \sum_{i=0}^{n-1} \mu_{33}(s_i) \quad (\text{C.45})$$

C.27 Hu moments

Let $\mu = (\mu_x, \mu_y)$ be the center of gravity, then the central inertia moments are computed as follows

$$m_{pq} = \sum_{i=1}^n (x_i - \mu_x)^p (y_i - \mu_y)^q, \quad \text{for } 0 \leq p, q \leq 3 \quad (\text{C.46})$$

In order to guarantee scale independence the moments are normalized:

$$\nu_{pq} = \frac{m_{pq}}{m_{00}^\gamma}, \quad \text{with } \gamma = 1 + \frac{p+q}{2} \quad (\text{C.47})$$

The seven Hu moments [47] are computed as:

$$f_{41} = \nu_{02} + \nu_{20} \quad (\text{C.48})$$

$$f_{42} = (\nu_{20} - \nu_{02})^2 + 4\nu_{11}^2 \quad (\text{C.49})$$

$$f_{43} = (\nu_{30} - 3\nu_{12})^2 + (3\nu_{21} - \nu_{03})^2 \quad (\text{C.50})$$

$$f_{44} = (\nu_{30} + \nu_{12})^2 + (\nu_{21} + \nu_{03})^2 \quad (\text{C.51})$$

$$f_{45} = (\nu_{30} - 3\nu_{12})^2 (\nu_{30} + \nu_{03}) \left[(\nu_{30} + \nu_{12})^2 - 3(\nu_{21} + \nu_{03})^2 \right] \\ + (3\nu_{21} - \nu_{03}) (\nu_{21} + \nu_{03}) \left[3(\nu_{30} + \nu_{12})^2 - (\nu_{21} + \nu_{03})^2 \right] \quad (\text{C.52})$$

$$f_{46} = (\nu_{20} - \nu_{02}) \left[(\nu_{30} + \nu_{12})^2 - (\nu_{21} + \nu_{03})^2 \right] \\ + 4\nu_{11} (\nu_{30} + \nu_{12}) (\nu_{21} + \nu_{03}) \quad (\text{C.53})$$

$$f_{47} = (3\nu_{21} - \nu_{03}) (\nu_{30} + \nu_{12}) \left[(\nu_{30} + \nu_{12})^2 - 3(\nu_{21} + \nu_{03})^2 \right] \\ - (\nu_{30} - 3\nu_{12}) (\nu_{21} + \nu_{03}) \left[3(\nu_{30} + \nu_{12})^2 - (\nu_{21} + \nu_{03})^2 \right] \quad (\text{C.54})$$

C.28 Normalized convex hull area

Let H be the convex hull around the gesture, then A_H denotes the area of the convex hull.

$$f_{48} = \frac{A_H}{w * h} \quad (\text{C.55})$$

C.29 Compactness

$$f_{49} = \frac{L^2}{A_H} \quad (\text{C.56})$$

Appendix D

Sonntag et al. Feature Set

The features described in this section are implementations based on the 14 features described by Sonntag et al. [96]. For this section we use the following notations:

A stroke is a sequence S of samples,

$$S = \{\vec{s}_i | i \in [0, n - 1], t_i < t_{i+1}\} \quad (\text{D.1})$$

where n is the number of recorded samples. A sequence of strokes is indicated by

$$\mathcal{S} = \{S_i | i \in [0, m - 1]\} \quad (\text{D.2})$$

where m is the number of strokes.

The centroid is defined as

$$\vec{\mu} = \frac{1}{n} \sum_{i=0}^{n-1} \vec{S}_i \quad (\text{D.3})$$

where n is the number of samples used for the classification, the mean radius (standard deviation) as

$$\mu_r = \frac{1}{n} \sum_{i=0}^{n-1} \|\vec{s}_i - \vec{\mu}\| \quad (\text{D.4})$$

and the angle as

$$\varphi_{s_i} = \cos^{-1} \left\{ \frac{(s_i - s_{i-1}) \cdot (s_{i+1} - s_i)}{\|s_i - s_{i-1}\| \|s_{i+1} - s_i\|} \right\} \quad (\text{D.5})$$

D.1 Number of Strokes

$$f_1 = |\mathcal{S}| \quad (\text{D.6})$$

D.2 Length

$$f_2 = \sum_{i=0}^{n-2} \|s_i - s_{i+1}\| \quad (\text{D.7})$$

D.3 Area

The area covered by the sequence of strokes \mathcal{S} is defined as the area of the bounding box that results from a sequence of strokes. We calculate the area of the convex hull A based on Graham's algorithm[38]

$$f_3 = \text{Area}(\text{Conv}(\mathcal{S})) = A \quad (\text{D.8})$$

D.4 Perimeter Length

The length of the path around the convex hull

$$f_4 = \|\text{Conv}(\mathcal{S})\| \quad (\text{D.9})$$

D.5 Compactness

$$f_5 = \frac{\|\text{Conv}(\mathcal{S})\|^2}{A} \quad (\text{D.10})$$

D.6 Eccentricity

Let a and b denote the length of the major or minor axis of the convex hull, respectively

$$f_6 = \sqrt{1 - \frac{b^2}{a^2}} \quad (\text{D.11})$$

D.7 Principal Axes

$$f_7 = \frac{b}{a} \quad (\text{D.12})$$

D.8 Circular Variance

Let μ_r denote the mean distance of the samples to the centroid μ . The circular variance is then computed as follows

$$f_8 = \frac{1}{n \cdot \mu_r^2} \sum_{i=0}^{n-1} (\|s_i - \mu\| - \mu_r)^2 \quad (\text{D.13})$$

D.9 Rectangularity

$$f_9 = \frac{A}{a \cdot b} \quad (\text{D.14})$$

D.10 Closure

$$f_{10} = \frac{\|s_0 - s_n\|}{f_4} \quad (\text{D.15})$$

D.11 Curvature

Let $\varphi(s_i)$ be the angle between the $\overline{s_{i-1}s_i}$ and $\overline{s_is_{i+1}}$ segments at s_i .

$$f_{11} = \sum_{i=1}^{n-2} \varphi(s_i) \quad (\text{D.16})$$

D.12 Perpendicularity

$$f_{12} = \sum_{i=1}^{n-2} \sin(\varphi(s_i))^2 \quad (\text{D.17})$$

D.13 Signed Perpendicularity

$$f_{13} = \sum_{i=1}^{n-2} \sin(\varphi(s_i))^3 \quad (\text{D.18})$$

D.14 Angles after Equidistant Resampling

For this feature we do an equidistant resampling with 6 line segments. The five angles between succeeding lines are considered to make the features scale and rotation invariant (normalization of writing speed).

$$f_{14} = \sum_{i=0}^4 \sin(\alpha_i), \sum_{i=0}^4 \cos(\alpha_i) \quad (\text{D.19})$$