Structured Handwritten Input for Dementia Classification

by

Gerardo Flores

A.B., Harvard College (2011)

Submitted to the Department of Electrical Engineering and Computer Science in Partial Fulfillment of the Requirements for the Degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2024

Authored By: Gerardo Flores
Department of Electrical Engineering and Computer Science
August 9, 2024

Certified By: Randall Davis
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted By: Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committee for Graduate Students

Structured Handwritten Input for Dementia Classification

by

Gerardo Flores

Submitted to the Department of Electrical Engineering and Computer Science
on August 9, 2024, in partial fulfillment of the
requirements for the degree of
Master of Science

Abstract

We explore the use of deep learning to score the Digit Symbol Substitution Test (DSST), a paper-and-pencil behavioral test useful in diagnosing Alzheimer's. We train a model to classify Alzheimer's based on the subject's responses to any one of the 108 queries in the test. We then combine predictions across the test to produce a new classifier that is considerably stronger. We also make an exensive search of architectures and optimization techniques that have proved useful in other settings. The ultimate result is a very strong classifier, with AUC for a response to a single question of 86% and for an overall patient of 97.25%.

Acknowledgments

I extend my deepest gratitude to my advisors, Prof. Randall Davis and Dr. Dana Penney, for their invaluable guidance and encouragement throughout this thesis. I am also thankful to the study participants and their families, whose courage underscores the importance of this research. Lastly, my heartfelt appreciation goes to my parents and friends for their unwavering support and belief in my abilities.

Chapter 1

Introduction

Alzheimer's disease is the most common form of dementia, but remains difficult to diagnose it early and therefore to treat it before it progresses. The Symbol Digit Substitution Test (SDST) was created as a simple and sensitive instrument to evaluate cognitive decline, but requires a lot of clinical judgment. Using a dataset of digitized pen strokes from SDST subjects with known diagnoses, we explore supervised machine classification of Alzheimer's disease.

The past fifteen years of advances in deep learning have created a range of new tools for processing complex sequence data, and we evaluate a range of architectural and optimization options, as well as the use of pretraining to boost performance. Overall, we find their performance against a baseline of linear models to be encouraging.

We begin with a review of the state of Alzheimer's disease, and its diagnosis.

1.1 Costs

The success of nutrition improvements and allopathic medicine have allowed more people to live long enough to experience dementia, and raised its impact and public salience (McKeown et al. 2020). Estimates of prevalence now run to 7 million americans (Alzheimer's Association 2024), and it is the seventh leading cause of death globally (Vollset 2024) and forecast to rise to 4th in the next twenty five years. One caregive describes her mother-in-law's experience:

> [She] wakes to find that she has aged 50 years overnight, that her parents have

> disappeared, that she doesn't know the woman in the mirror, nor the people who claim to be her husband and children, and has never seen the series of rooms and furnishings that everyone around her claims insistently is her home.(Gillies 2011)

Financially, the total cost of caring for Alzheimer's patients in the US was estimated to be in the range of $200 to $450 billion depending on how informal care is valued (Nandi et al. 2024).

1.2 Diagnosis

As the understanding of mental illness progressed in the late 19th and early 20th centuries, varieties of early onset dementia began to be more thoroughly classified. Alzheimer's, which is by an order of magnitude the most common of them, was first clinically described in 1906 following an autopsy of a 51 year-old patient's brain, and came to be understood as presenile (early onset) dementia (Keuck 2018). Alzheimer himself noted that his patients condition was accompanied by what he labeled "amnestic writing disorder" (Maurer, Volk, and Gerbaldo 1997), but autopsy was the only universally accepted test of Alzheimer's for about a century after his work.

Today, CMS reimburses a $3,000 PET scan to look for amyloid plaques, and blood tests are beginning to enter the market, but blood and cerebrospinal fluid tests are still considered highly unreliable. Diagnosing cognitive impairment, and particularly dementia, is currently quite difficult to do given the desire of medicine to have a high confidence, objective, chemical or radiological assessment. However, lack of obvious biomarkers or simple structural changes has made this quite difficult. Today, there are PET scans available to detect Tau protein plaques in the brain (believed to be implicated in Alzheimer's), but due to the short half-life of the radio tags, they cost $20,000 per administration. Other options like blood or even cerebrospinal fluid levels of these proteins are not conclusive. An estimated 90% of Alzheimer's patients do not have these assessments. In any case, the final assessment is still done face to face by a trained clinician (Knopman and Hershey 2023).

The most used and studied instrument of cognitive decline today is the Mini-Mental State Examination (MMSE), which typically takes 7-10 minutes to administer. For Dementia, it's estimated to have about 90% sensitivity and specificity. A briefer instrument like

the Clock Drawing Test (CDT), which can take under a minute, is not quite as good, more like .85 / .85. However, performance for MCI is considerably poorer. For MMSE, it's more like .5 and .75, respectively, and for CDT, more like .65 / .65 (Patnode CD 2019).

The attributes of Alzheimer's are in some ways most specific early on. In particular, early Alzheimer's patients show marked short-term memory decline before they show other classic deficits in cognitive processing. This suggests that a Digit Symbol Substitution Test, which focuses on memory, can be more productive than other very short methods, like the clock drawing test (Kumar A 2022). It's already been known for a few decades that DSST can measure both cognitive deficits and recall issues, and can be used to discriminate between Alzheimer's and Parkinsons (Demakis et al. 2001).

Screening is valuable because it opens treatment avenues, but also simply intrinsically to confront the fear of the unknown.

> Most of what we forget is not a failure of character, a symptom of disease, or even a reasonable cause for fear—places most of us tend to go when memory fails us. We feel worried, embarrassed, or plain scared every time we forget something we believe we should remember, or would have remembered, back when we were younger. We hold on to the assumption that memory will weaken with age, betray us, and eventually leave us.(Genova 2021)

1.3 Alzheimer's Treatment

For a century, evidence on disease-modifying clinical responses to Alheimer's has been weak. In the past few years, a few medications have finally been approved to slow the disease course, but only for very mild cases. This makes early detection and treatment more important than ever.(Belder, Schott, and Fox 2023)

However, diagnosis is still based on specialist clinician interactions (Knopman and Hershey 2023). These interactions are enormously expensive and also rate-limiting. Any tools that enable general practitioners to diagnose the disease earlier and more routinely would be of enormous value.

1.4 Alzheimer's Handwriting

Since Alois Alzheimer's "amnestic writing disorder" (Maurer, Volk, and Gerbaldo 1997), it has been known that there are changes in handwriting that accompany Alzheimer's. However, unlike Parkinson's distinctive micrographia, these signs of Alzheimer's require a great deal of specialized experience. A meta analysis of studies of handwriting shows that longer delays between strokes, in particular, are indicative (Werner et al. 2006), but most simple features like size and average velocity appear to vary by setting (Fernandes et al. 2023).

A compelling avenue would be to take the rich input data available from filling out forms on an ipad to predict the classification. If this can be brought to parity with a trained specialist, it can significantly improve the diagnostic capability of a typical general practitioner or gerontologist, who lacks the neuropsychological training and indeed the time to administer the exam. On an engineering level, this requires modeling techniques which can use rich data sources with relatively little hand-tuning.

1.5 Deep Learning

The past fifteen years have seen a revolution in statistical learning through the exploration of massively overparameterized models. It had been understood for a few millenia that parsimonious models tended to generalize better, a point described in the context of regression by the use of information criteria (Akaike 1974). It more recently came to be understood that once the number of parameters began to exceed the number of data points, the quality of out of sample generalization began to increase again (Belkin et al. 2019). However, fitting large models has traditionally been quite difficult, because they have tended to be non-convex, and analytically intractable.

A few major developments combined to change this. First of all, the continued speedup of processing, and particularly the reuse in machine learning of hardware parallelism tools first developed for graphics, vastly sped up processing of matrix math. Second of all, automatic differentiation (Wengert 1964) reached a maturity where practical libraries became widely available and useful. This, coupled with a willingness to use simpler if less efficient first order methods based directly on the gradient (Richardson 1911), vastly reduced the implementation difficulty of gradient descent methods, which are among the few practical

tools that work in extremely high dimensional optimization. Third, there was a realization that model architecture and training procedures should be designed to reduce the noise of individual gradient estimates by applying huge numbers of updates to each parameter. This takes the form of both parameter sharing, in architectures like CNNs LeCun et al. 1989, and various techniques to get more updates from the same data, including data augmentation and next-step pretraining.

Overall, there is a wide range of new architectural options as well as optimization techniques for accelerating Richardson Iteration / Gradient Descent. We evaluate them extensively to produce a good classifier for this particular dataset, and also to provide a guide for future work.

1.6 Objectives

This thesis aims to explore deep learning architectures and optimization methods to classify Alzheimer's disease based on handwriting data from the digit symbol substitution test. In particular, the decomposition of the overall test into 108 questions and answers allows more efficient use of the data to fit large models, and the combination of 108 separate predictions makes for more accurate overall classification. However, the pen strokes in each answer still form a complex sequence of data, and a range of technical approaches is explored.

Chapter 2

Related Work

2.1 Handwriting

The measurement of cognitive deterioration by examining writing is well established. However, there are obstacles to routine use, including the time taken to evaluate the results, and the difficulty of standardizing interpretation (Moetesum et al. 2022). In recent years, emphasis in the field has moved from what (Moetesum et al. 2022) terms visual to procedural analysis, emphasizing additional information such as pressure and time to completion. In most state of the art applications, these sequences are post-processed (in narrow temporal windows) to acquire information about dynamics, and dynamics are then aggregated through summary statistics.

More recent work has used advanced DSP techniques, including cepstral analysis (Nolazco-Flores et al. 2021). However, cepstral analysis is based on assuming that each impulse within the writing has the same shape, so this may boil down to laplacian edge detection in the log spectrum. There has also been work with fractional derivatives (Mucha et al. 2023). However, sequence modeling has still not substantially taken off (Ayaz et al. 2023).

In the specific context of cognitive impairments, (Prange and Sonntag 2022) claims roughly 85% specificity /sensitivity from using a grab bag of 176 handwriting features culled from the literature, without specifically tailoring for what is being drawn in the test. This is promising, but since hand-implementing 176 features correctly can be daunting, suggests that a deep-learning based approach may be more powerful and more maintainable.

Another prominent recent example has been (D'Alessandro et al. 2023), which studies velocity alone, and decomposes velocity timeseries for a single connected pen stroke into a

sum of individual accelerations, each with an offset log-normal shape, which is unimodal, and skewed to the right. They claim to achieve accuracy in the region of about 75% in classification.

2.2 DSST

The attributes of Alzheimer's are in some ways most specific early on. In particular, early Alzheimer's patients show marked short-term memory decline before they show other classic deficits in cognitive processing. This suggests that a Digit Symbol Substitution Test, which focuses on memory, can be more productive than other very short methods, like the clock drawing test (Kumar A 2022). It's already been known for a few decades that DSST can measure both cognitive deficits and recall issues, and can be used to discriminate between Alzheimer's and Parkinsons (Demakis et al. 2001), and it is arguably the most commonly used neuropsychological assessment tool (Jaeger 2018).

Of particular interest have been studies in the past couple of years using the Digit Symbol Substitution Test. For example, (Campitelli et al. 2023) implements the DSST on a digital platform, and finds that (# right in 2 minutes) has correlation of about .6 with the longer and more costly Repeatable Battery for the Assessment of Neuropsychological Status.

Similarly, (Williamson et al. 2022) modifies the DSST to match images of medications with days of the week in order to try to reduce reliance on familiarity with neuropsychological evaluation, and administers it online, and finds that (# correct in 2 minutes) has about .4 correlation with the MMSE.

In addition, (Lesoil et al. 2023) similarly implements DSST on a digital platform and finds (#correct in 2 minutes) to have correlation with .7 with the MMSE.

Moreover, (Andersen et al. 2021) studied time taken to think and write in different segments of the test, and whether changes during administration were predictive of MMSE (statistically significant differences), as well as decline in performance on followup evaluations (also statistically significant), in an international sample.

2.3 Handwriting on DSST

(Cook 2023) explored implementation of a DSST that captures pen strokes, and administered the Montreal Cognitive Assessment (MoCA) to 17 patients, but did not evaluate the correlation between the two instruments. Previous work from the same group, with an older version of the test, hand-engineered a set of features and found an AUC for AD vs Healthy, PD vs Healthy, and AD vs PD in the range of about .97 (Huang 2017). Published evaluations of Sainte-Cerveau, a superset of the DSST, claims 97% sensitivity and 91% specificity for AD.

However, there has not been work directly feeding the pen stroke data into a deep learning model and training end-to-end. This would allow the automatic discovery of features that are most predictive of the outcome, and a richer understanding of the data.

Chapter 3

Data

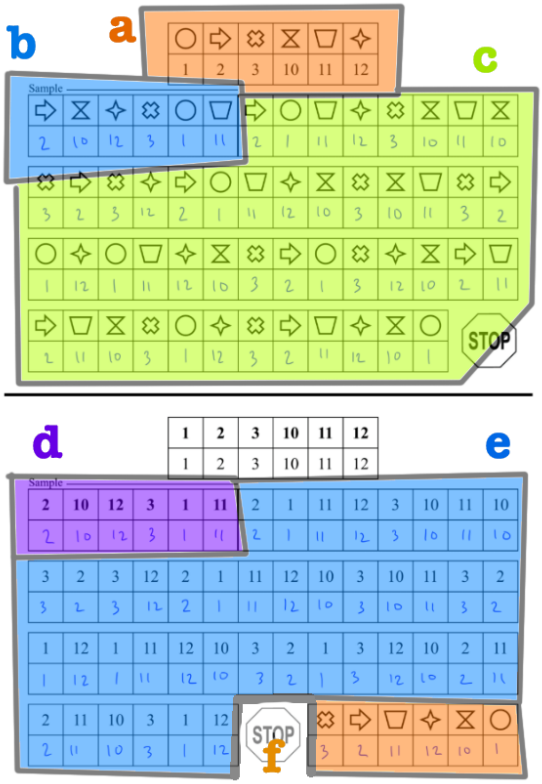

Figure 1: The SDST. (a) the answer key (b) untimed translation (c) timed translation (d) untimed copying (e) timed copying (f) untimed pure recall

3.1 Symbol Digit Substitution Test (SDST)

The SDST is a refined version of the DSST, designed to be easier to score, and to increase informativeness in the context of dementia. Section [a] (the answer key, see Figure 1) is similar to the DSST answer key, except the sy mbols have been made simpler and more distinct, and only the digits "0", "1", and "2" are used. A moment's reflection on 4/9, 1/7, 3/8 will suffice to show why this simplifies evaluation. Section [b] (the untimed translation) and section [c] (the timed translation) are roughly the same as the DSST, although [c] has a more uniform distribution of the tasks than the DSST.

Sections [d] and [e], which are not found in DSST, are similar to [b] and [c], except there is no translation involved. The intent is to provide a control group to isolate and control for effects based on motor problems alone, which explain a significant fraction of the variation in test outcomes - one study found 72% of the difference between younger and older patients (Lezak 2004). Finally, section [f] asks the patient to recall what was in the translation key they saw a few minutes prior. For reference, the author, a second year student being of as sound of mind and body as a person might be and choose to enter a doctoral program, found this difficult.

Finally, the SDST repeats the same exercise a second time, in order to quantify improvements in the task based on associative learning.

Meta Category	Characteristic	Count
Age	<65	44
	65-74	20
	75-84	17
	85+	2
Sex	F	55
	M	28
Race/Ethnicity	White, Non-Hispanic	75
	Black	7
	Hispanic	1

Table 1: Demographic information for the 83 patients in the study.

3.2 Demographics

The data used in this thesis was obtained from recordings of pen strokes by 83 patients completing the SDST using a digitizing ballpoint pen. The tests were administered, and ground truth diagnostic labels assigned by Dana Penney, director of Neuropsychology at Lahey Clinic, along with collaborators. ask Randy about the details and appropriate citation here Patients were mostly caucasian, reflecting demographics at Lahey, and predominantly female and under the age of 65 (see table 1).

TODO 1 ask Randy abo details and appr citation here

We use 90% of the data for training and tuning, and hold out 10% of the sample for evaluation. Since this produces a small test set, we also run a robustness check in which we train on 60% of the data and evaluate on 40% of the data. The results are similar, suggesting that good performance is not simply an artifact of small sample size.

3.3 Preprocessing

The use of deep learning sequential models reduces the need for careful feature engineering. Nevertheless, there were two important challenges. First, correctly assigning the pen strokes to the question they were intended to answer allows more efficient use of data. Second, some simple processing of position and time to create dynamical features makes the baseline linear model more competitive, and allows feature ablations to give more insight into what aspects of handwriting the model is using.

3.3.1 Assigning Pen Strokes to Questions

First of all, the questions are each answered in a separate box (see 1). However, subjects' pen strokes don't always stick within a given box. A pretty straightforward solution to this is to pool all the positions at which the pen was observed, and assume that this density is roughly a mixture of gaussians, with the gaussian means fixed in a grid. We can then estimate the grid location using maximum likelihood, a convex optimization problem that is readily solved by standard methods (e.g. those provided by scipy.optimize.minimize).

In practice, I found that the exact edges were still quite difficult to determine. Unfortunately, no single offset provided a clear separation of strokes; it seems that subjects were crossing over the edges of the boxes that separate answers. Of course this is no problem for a human clinician, but an algorithm needs a decision rule. After manual inspection, we restorted to the oldest machine learning method: a graduate student inspected the data and manually created the decision tree shown in Algorithm 1.

Algorithm 2 The decision rule for mapping pen strokes which cross box boundaries.

```
procedure MapStroke(stroke)
    if at least 75% of stroke is in one box then
        map stroke to that box
    else if ignoring points close to the edge, at least 75% of stroke is in one box then
        map stroke to that box
    else if stroke is much closer to preceding than succeeding stroke then
        map stroke to the same box as the preceding stroke
    else if stroke is much closer to succeeding than preceding stroke then
        map stroke to the same box as the succeeding stroke
    else
        map stroke to the box where it starts
    end if
end procedure
```

Once we've mapped the pen strokes to the question they were intended to answer, then we readily infer what the correct answer was, and what type of memory and processing were involved. For example, the correct response in the upper left box in the whole response is "1", and it's a simple copying task.

3.3.2 Pen Stroke Dynamics

After both speaking with clinicians and examining the literature (e.g. (D'Alessandro et al. 2023)) we find that simple dynamics (the profile of stroke speed, in particular) is useful for building linear models. Since we're doing this for a baseline anyway, we use it as an input sequence for all the models.

The obvious features borrowed from physics would be time, position, velocity, acceleration. We add jerk, the lesser known third derivative of position, which measures the steadiness with which force can be applied by a subject. Moreover, to maintain translational and rotational invariance (conjectured to be important since subjects don't always find the center of the boxes provided for answers), we feed the log magnitude of these quantities into the model, rather than their x and y components.

However, the interaction between the directions of velocity and acceleration is also important. Longitudinal acceleration, along or against the direction of velocity, represents a patient speeding up or slowing down a stroke. Centripetal acceleration, orthogonal to motion, reflects the curvature of the stroke. In this case, the standard measure in the literature is curvature, a closely related quantity - see table for details.

Metric	Normalization
Time	$t - t_0$
Velocity	symlog1p $K\|\frac{dx}{dt}\|$
Acceleration	symlog1p $K\|\frac{d^2x}{dt^2}\|$
Jerk	symlog1p $K\|\frac{d^3x}{dt^3}\|$
Change in Speed (Longitudinal)	symlog $K\frac{d}{dt}\|\frac{dx}{dt}\|$
Curvature (Centripetal)	symlog $K\frac{\|\frac{dx}{dt}\times\frac{d^2x}{dt^2}\|}{\|\frac{dx}{dt}\|^3}$

Table 2: The 6 dynamical features used as inputs to linear and deep models. For explanations of symlog and symlog1p , see 3.3.3.

3.3.3 Dynamical Range

Because these dynamics have a wide dynamic range, we log-scale them. However, since some of them can take zero or negative values, we use a modified function sometimes referred to as the signed or symmetric logarithm. I use here the name 'symlog' which is typical in python packages, particularly the matplotlib library. See the figure for intuition, and then the equations below for the details.

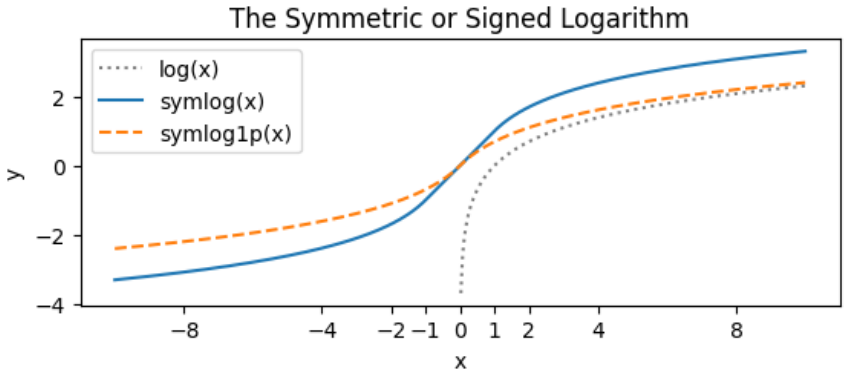

Figure 2: Two implementations of the symmetric or signed logarithm function.

The simplest implementation asymptotically matches $\text{sgn}(x) \log |x|$ for large values of x.

$$\text{symlog1p } (x) = \begin{cases} \log(1+x) & x > 0 \\ -\log(1-x) & x < 0 \\ x & \text{otherwise} \end{cases}$$

Although the scaling near zero is easy for an algorithm to handle, it's a bit more annoying for a human to understand. An alternate implementation is simply linear between -1 and 1, but then remains 1 unit larger than the logarithm for large values. We used this in practice to simplify debugging of quantities that were meant to take both positive and negative values, though the difference is not really significant to the models.

$$\text{symlog } (x) = \begin{cases} 1+\log(x) & x > 1 \\ -1-\log(-x) & x < -1 \\ x & \text{otherwise} \end{cases}$$

Chapter 4

Methods

The overall goal of this project is to explore methods for classifying Alzheimer's based on pen strokes on the SDST. On a technical level, we fit the models using gradient descent on a loss function, which is the binary cross entropy loss for the classification task.

$$\text{loss}(y, \widehat{P}(y)) = \sum y \log \widehat{P}(y) + (1 - y) \log(1 - \widehat{P}(y))$$

For a large and complex model this optimization problem can be quite ill-conditioned. However, large and complex models have shown radically better performance than smaller, simpler models in other domains (particularly vision and language), so the purpose of this thesis is to explore whether they can also radicall improve performance in Alzheimer's classification. To test this proposition, we experiment with various modern optimization techniques for accelerating gradient descent on these larger models.

In addition to the difficulty of optimization methods, larger models have shown to require larger amounts of labeled data. By fitting a prediction on each of 108 answers for each of the 83 subjects, we expand the number of labeled examples for training by about two orders of magnitude. However, there are still on the order of about 100 data points per answer. In many domains, training machine learning models to predict the local structure (the next word, the next pixel, or in our case the speed and direction of motion a few tens of milliseconds later) has shown to produce more powerful models. When these are then fine-tuned for use in classification tasks, they sometimes outperform models directly trained on classification.

Dimensionality Reduction	Example	Pro	Con
Raster	2D CNN	Doesn't require digital pen	Data loss
Scalar	Logistic Regression	Fast, well understood	Not very sensitive
Symmetric	1D CNN	More expressive	No pretrain
Causal	Transformer	Next step pretrain	Less flexible

Table 3: High architectural approaches

4.1 Architecture

A central challenge of the pen stroke data is that it's originally formatted as a variable-length list of (x,y) pairs. Even when we preprocess the features to get information about dynamics, we need an approach to reducing the dimensionality if we want to train a classifier.

At a high level, these may be divided into scalar approaches (e.g. a logistic regression based on maximum velocity, acceleration, etc), raster image based approaches (e.g. a CNN on a 28x28 image), and sequence approaches (e.g. a Transformer on a 128 timestep sequence of values). Sequence approaches can be further divided into those that treat past and future symmetrically (e.g. a 1D CNN), and those causal approaches which only use data up to a point in time. Causal approaches have proven particularly successful in natural language processing, where the prediction of the next token has produced the recent revolution in large language models, and produced much better tools for classification.

4.1.1 Raster

We can take the x,y coordinates and convert them into a raster image of shape $BxWxH$, where B represents the batch (training many examples at once for efficiency), and W and H are height and width, which are set to 28 for compatibility with the standard MNIST task (Deng 2012). We then train a 2D convolutional model (LeCun et al. 1989) to predict the patient status based on this pixel data, which takes about 40s.

4.1.2 Scalar

As a baseline for what can be done with the preprocessed dynamical features, we simply take both the min and max values of all dynamics for the entire stroke, producing data of shape BxC where B represents the examples, and C represents the "channels" (min and max values of features over the stroke). We then run a logistic regression using these inputs.

This includes temporal information that's left out of the raster model, and is fast (about one second to train) and statistically very well understood, but ignores layout and ordering.

4.1.3 Symmetric (1D)

In this setting, we arrange the data to be of shape $BxLxC$, where L is the length of the longest pen stroke data, and C are the 6 dynamical features. For reasons of practicality, we truncate stroke length to 128 tokens, which captures 97% of the data, and we pad the sequences with zeros to get the same length in a given batch. We then run a one dimensional CNN on this sequence data.

We compare a variety of different approaches to this CNN.

Model	Training Time	Motivations
Baseline	7s	Well understood, fast and easy to optimize
À Trous	12s	Longer connections improve performance, but slower to train
Butterfly	16s	Mathematically more elegant than à trous , but may not match hardware
FFT	15s	Elegant, sometimes more effective on time series

Table 4: 1D CNN approaches

Background: the Feedforward Block

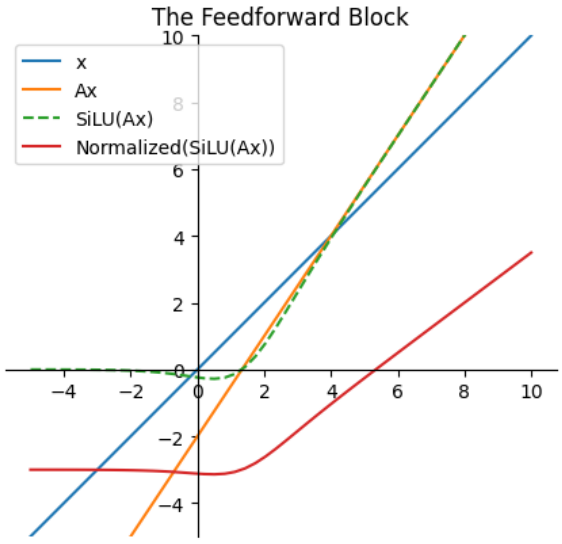

Figure 3: The Feedforward Block

Although the idea of deep learning is predominantly to compose similar, nearly lin-

ear functions with each other to produce more complex functions, the details of standard practice are worth explaining.

In a standard feedforward block, the inputs are multiplied by a matrix to produce outputs of a different size. Then the outputs are "rectified" to cap negative values near zero. Then rectified outputs are normalized so that in any given batch of training samples, the outputs will have mean zero and variance 1 (Ioffe and Szegedy 2015). A new linear transformation is composed with this, just rescaling and shifting the output distribution, but maintaining the stability from batch to batch. This stability of distribution greatly improves the conditioning of the optimization problem, which makes training faster.

There are various rectifying functions in current use, but we tested a variety and found the best results with the SiLU function (2017).

Baseline CNN

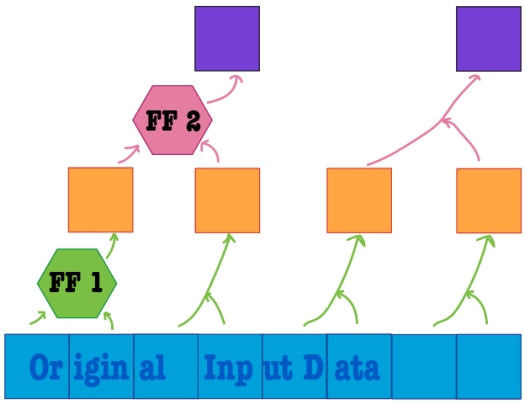

Figure 4: The Baseline CNN

The key idea of a basic CNN is to cut the sequence into non-overlapping windows, and run the same feed-forward neural network over each window, producing a shorter output sequence LeCun et al. 1989. This is done repeatedly until we have a single output, which is fed to a scalar classifier. In some cases, at some scale we take a maximum along the sequence over the output channels. This allows better handling of varying-length inputs.

Because the same feedforward block is used at each point in the sequence, each parameter is updated by gradients from many more data points, which makes for far more stable training.

A search of the design space found best performance came from cutting into pieces of size 2 in the first layer, then 2, then 16 and then the maximum value of each output channel is kept, producing a sequence of length 1.

À Trous

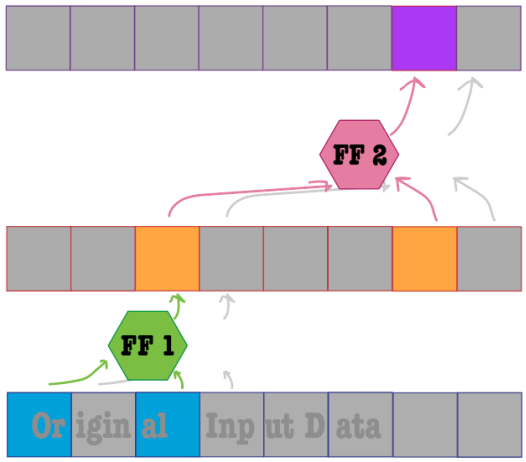

Figure 5: The À Trous CNN

The key idea of the à trous CNN is that our input window doesn't have to sample contiguous points in the sequence (Chen et al. 2018). If we instead sample points with a longer spacing, we can more easily mix information on different scales. This means the sequence length doesn't decrease as we stack more layers, which slows down the model, but can also produce better classifiers.

FFT

Rather than carefully pick a strategy to sample the sequence at different spacings, we can simply take a fast fourier transform of the sequence, and keep the first N lowest frequency components. Then we can process this (much shorter) sequence with a standard CNN. This idea was popularized in the context of sequence attention for transformers by (Lee-Thorp et al. 2021). While it didn't prove useful for transformers (because it cannot be made causal, see below), it is useful for symmetric treatment of 1d sequences.

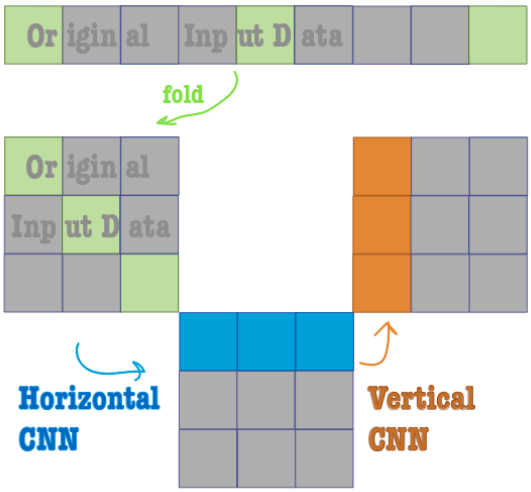

Figure 6: The Butterfly CNN

Butterfly Matrix

A generalization of many schemes for processing data at different scales, including the à trous convolution and the FFT, is the butterfly matrix. It begins by folding the 1d input into a 2d grid, and then operates on that 2d grid with 2d convolutions. This is mathematically the same as the à trous convolution, but conceptually elegant, and can be faster if the CUDA compiler used on the hardware in question is optimized for it (Dao et al. 2019).

Performance Comparison

In fact, the FFT gives the best discrimination of the 1d models. AUC-ROC, a well known metric, can be interpreted as the probability that given one response from a person with alzheimer's, and one from a person without, the model would correctly guess which was more likely to have Alzheimer's. A higher AUC-ROC is better. Brier Score, the mean-squared error of the probabilistic prediction, gives more insight into how well calibrated the probabilistic predictions of the model are. It can be interpreted as the average cost of misclassification if the base rate of the positive classes varies uniformly between 0 and 1. This linear / decision-theoretic meaning is less important, because we intend to combine predictions across many questions. A lower Brier Score is better.

Model Type	AUC-ROC	Brier Score
1D CNN	0.789 ± 0.019	0.215 ± 0.017
À Trous	0.811 ± 0.010	$\mathbf{0.206 \pm 0.004}$
FFT	$\mathbf{0.823 \pm 0.024}$	0.213 ± 0.018
Butterfly	0.765 ± 0.015	0.222 ± 0.004

Table 5: 1D model types and their evaluation metrics on holdout sets. Standard deviation in parentheses, tuned on 10 runs.

Model	Training Time	Filter Type	Motivations
TCN	62s	Finite Impulse Response	Fast, simple, requires few tricks.
TCN (custom)	16s	Finite Impulse Response	Same as TCN, but hand-tuned
LSTM	40s	Infinite Impulse Response	Does well on short sequences.
Transformer	68s	Embedding Space	Difficult, very good with big data.

Table 6: Causal sequence modeling approaches

4.1.4 1D Causal (Next Step Pretraining)

Rather than treat the past and future symmetrically, we can design models which can only react to the past. These so called causal models can be used in real time, but they can also be trained to predict future values of the sequence. That approach creates two orders of magnitude more labeled examples (one at each time step), and can in some cases significantly improve the model performance by using more generalizable features (Dai and Quoc V Le 2015).

There are three basic types of these models. The first two (TCN and LSTM) are similar to those known from signal processing: finite and infinite impulse response, respectively. The third (Transformer) learns a connectivity graph between elements of the sequence, only loosely based on their position in the sequence, and then runs calculations using proximity in that graph rather than time. Finally, because the TCN approach is the simplest, and we trained models on non-standard hardware, we also hand-wrote a TCN implementation to see if we could get better performance.

Finite Impulse Response: TCN

TCN is like à trous , except that the filters are aligned so that only past data is used in calculating at any point in time (Lea et al. 2016). Since we cannot simply downsample the data, we have to use the à trous convolution to sample more distant points in the past. Since

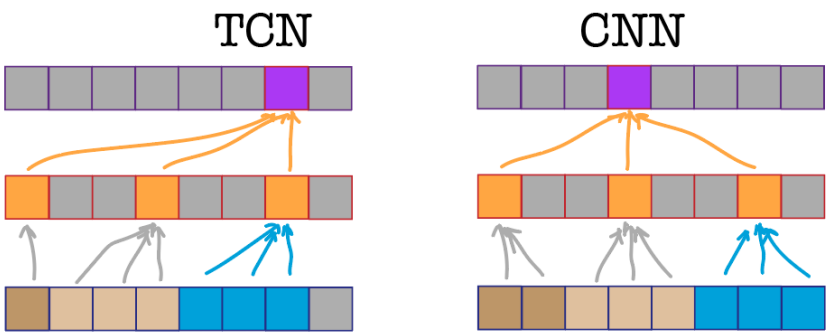

Figure 7: Temporal Convolution Networks vs normal Convolutional Neural Networks

the fields are aligned, the next time point will not appear in the inputs to the regression, and it can be used to predict the next step.

For classification purposes we can simply max-pool the outputs at all time steps again.

Classification-Optimized TCN

Since the next-step version of the TCN does not shrink the sequence, training is quite slow. A simple optimization approach to speed up classification training might be to decimate the outputs that will not be reused for the final result. This was hand-implemented, and performed comparably to an off-the-shelf TCN implementation, while speeding up by a factor of 4 or so.

Infinite Impulse Response: LSTM

An ironically older approach is the Recurrent Neural Network (RNN), which learns a (mostly) linear dynamical system whose inputs are the features of the data, and whose outputs are the next step's features. This never really scaled up to very long sequences, and is no longer used as much for natural language, but is still a viable approach for this kind of data.

We use the Long Short-Term Memory (LSTM) variant of the RNN (Hochreiter and Schmidhuber 1997), which keeps an additional nonlinear hidden state, which receives inputs with time-varying intensity. Although later work found that GRU (Chung et al. 2014) does as well as LSTM with fewer parameters, the community never really saw a strong reason to change, and for our purposes the more popular LSTM is fine.

Unlike the TCN, in this case for classification we will simply take the final hidden state of the dynamical system as an input to our classifier.

Transformers

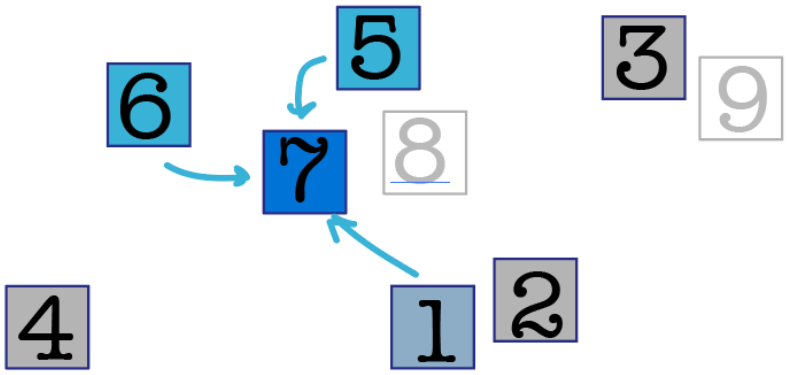

Figure 8: Attention in a transformer model. Blue represents points in the input sequence that strongly affect the output. White represents points in the future, which are not used in the calculation. Order in the sequence (the numbers) is only a part of what determines which points are connected.

In the last 10 years, deep sequence modeling has largely been taken over by transformers, whose receptive field is based on similarity in an embedding space, rather than position in the 1D sequence of input tokens (Vaswani et al. 2017). The strength of the connections between tokens (known as attention) is still generally partly based on distance, with an exponentially decaying strength of connection (Press, N. A. Smith, and Lewis 2022).

However, CNNs and RNNs provide a tremendously strong inductive bias towards the effect of sequence elements on each other based on position in the sequence, whereas transformers largely need to learn representations of the proximity graph of elements by embedding them into a higher dimensional space. This requires better optimization techniques, and generally requires orders of magnitude more data to pay off. The back of the envelope suggests the number of data points here is probably only slightly too low, so it's worth testing the performance of these models. Although at this point clearly better for natural language (Radford et al. 2019), it's less obvious for low dimensional, small data problems.

However, the first step of embedding the low-dimensional, continuous features we've engineered into a high dimensional space can be particularly difficult to learn. One technique

used in such circumstances is to use an untrained CNN layer with random coefficients, and then pass the output through a sine transform. This approach can be more stable and therefore faster to train than the normal CNN layer, and is known to be able to approximate kernel functions between inputs(Rahimi and Recht 2007). We tested both approaches, and used the sine transform in the final model.

Performance Comparison

In fact, TCN seems to have the best performance, although it's hard to distinguish from LSTM. Transformer really doesn't seem to have enough data to deliver better results.

Type	AUC-ROC	Brier Score
TCN (custom)	0.840 ± 0.007	0.199 ± 0.004
LSTM	0.833 ± 0.009	$\mathbf{0.194} \pm 0.004$
TCN	$\mathbf{0.855} \pm 0.005$	0.195 ± 0.003
Transformer	0.821 ± 0.012	0.198 ± 0.004

Table 7: Performance without pretraining. Standard deviation in parentheses, tuned on 10 runs.

Pretraining

Here we report the accuracy of the pre-trained model in forecasting future dynamics, using root mean-squared error as a metric (lower is better). We also take that pre-trained model and fine-tune it on the classification task, and report the AUC-ROC and Brier Score for that model. Surprisingly, pretraining basically doesn't improve any of the models' performance much (maybe the LSTM, but only very slightly). The transformer delivers substantially better results in next-step prediction, but they don't transfer to classification.

Type	RMSE	AUC-ROC	Brier Score
TCN (custom)	1.096 ± 0.001	0.834 ± 0.012	0.201 ± 0.004
LSTM	0.938 ± 0.116	$\mathbf{0.849} \pm 0.009$	$\mathbf{0.190} \pm 0.004$
TCN	1.929 ± 0.134	0.824 ± 0.020	0.210 ± 0.009
Transformer	$\mathbf{0.759} \pm 0.010$	0.806 ± 0.004	0.205 ± 0.002

Table 8: Performance in next-step prediction, and classification performance of pretrained models. Standard deviation in parentheses, tuned on 10 runs.

4.2 Optimization

Unlike convex problems, the best techniques for optimizing deep neural networks are not theoretically clear. A wide range of techniques are discussed in the literature, but some organization is possible. The major categories are:

- Discretization and Conditioning: carefully following infinitesimal gradient flow to the optimal parameters of an enormous model is impractical. We explore discretization and techniques for handling ill-conditioned problems.
- Scheduling: the learning rate is a critical hyperparameter that determines the size of steps that is taken. However, changing the size of the learning rate during training can produce better results. We explore a few of the most popular techniques.
- Gradient modifications: traditionally gradients are evaluated at the current parameter values, but there are various techniques to evaluate gradients at carefully chosen nearby points, or otherwies directly modify the update to the model parameters.

4.2.1 Discretization and Conditioning

The workhorse of deep learning is stochastic gradient descent, in which a small batch of data samples are collected, the classification is made, and then the weights are perturbed in a direction that maximally moves the predictions in the right direction.

However, there are several variations on this theme. We'll begin with the simplest.

Gradient Flow

$$\frac{dx}{dt} = -\nabla f(x)$$

In theory, we can solve a convex problem by simply following the direction of steepest descent until we find a minimum. The catch here is that in the limit that the step size becomes infinitesimal, the number of steps becomes infinite. In fact, however, the loss function we're training is based on randomly sampled training data, so finding the exact minimum may not be useful.

In practice, we can't run this forever, so we approximate it using a Runge-Kutta 4 solver, whose error goes like the fourth power of the learning rate.

Gradient Descent

$$x_{t+1} - x_t = -\eta \nabla f(x_t)$$

This is the fundamental workhorse of deep learning. We pick a step size (learning rate), and we advance that far in the direction of steepest descent. For problems with small number of parameters, we can actually approximate the local curvature by a parabola, and use that to guess how far to step forward. However, calculating the hessian is quadratic in the number of parameters, so in practical deep learning, these second-order methods are never usable.

A central problem is to find a decent step size for an approximation here. While there are clever ideas that work pretty well for convolutional architectures (L. N. Smith 2017), they don't work for all architectures. Often a coarse grid search on a log scale, using single-epoch results, followed by more careful gaussian-process based exploration with full runs will produce notably better results.

SGD

$$x_{t+1} - x_t = -\eta \nabla f(x_t, \text{batch})$$

In practice, we rarely have enough working memory to calculate the loss for the entire set of data at once. Given the linearity of derivatives, we could run through the whole data set, add up all the gradients, and then update the parameters once.

But in practice this is usually wildly inefficient. Instead, updating parameters after each segment of the data set is processed allows much faster convergence. Stochastic gradient descent now has two free parameters: the step size (learning rate), and the size of the batch of updates which are applied together simultaneously. In general, it is believed that the learning rate should vary inversely with the batch size (Fu et al. 2023).

Note that the model chosen to illustrate these principles is quite small, and shrinking the batch sizes produced only a very small speedup. However, in general, the difference can be significant.

The choice of batch size 1000 is somewhat arbitrary here.

SGDM

$$y_{t+1} - y_t = -\beta y_t + \nabla f(x_t, \text{batch})$$

$$x_{t+1} - x_t = -\eta y_t$$

For a variety of reasons, smoothing the parameter updates may be better than directly taking the direction of steepest descent. Principal among them:

- updating the parameters after every single batch of gradients is calculated is very noisy. Rather than wait to apply them all at the same time, we can smooth out that noise with an exponential moving average.

- given the poor conditioning of all optimization problems in reality, a step size big enough to make progress on the directions with low curvature will generally oscillate in the directions of high curvature, and this oscillation will slow training

- in fact without momentum, this oscillation will typically not only waste time, but actually cause the solver to diverge. This would constrain us to very low learning rates, which is slow.

- faster learning rates appear to produce better generalization.

However, this requires twice as much memory as SGD, because during training there's an additional shadow momentum parameter that needs to be stored for every single parameter of the model.

Signed Momentum

$$y_{t+1} - y_t = -\beta y_t + \nabla f(x_t, \text{batch})$$

$$x_{t+1} - x_t = -\eta \cdot \text{sgn}(y_t)$$

The atrocious conditioning of these optimization problems means that even with smoothing, we often get gradients that are gigantic in one direction or another. One appealing solution is gradient clipping, where no direction is allowed to have more than a given size update. An even simpler solution is to simply have fixed-magnitude updates for each coordinate, and use the sign of the gradient only to inform this.

This update rule has been found to be effective for very large language models. But for most smaller applications, the amount of noise this creates is untenable, and the models are essentially untrainable.

Adam

A more complex approach would be to estimate a diagonal preconditioner for the problem. In effect, Adam keeps a running estimate of the curvature of each parameter, based on the variance of the gradient. It then rescales updates based on that curvature. (Kingma and Ba 2017)

$$y_{t+1} - y_t = -\beta_1 y_t + \nabla f(x_t, \text{batch})$$

$$v_{t+1} - v_t = -\beta_2 v_t + \|\nabla f(x_t, \text{batch})\|^2$$

$$x_{t+1} - x_t = -\eta \frac{y_t}{\sqrt{v_t} + \epsilon}$$

Performance Comparison

Not very surprising: Adam comes out the winner, as usual. Signed Momentum is surprisingly close, though. Because it doesn't track the curvature of the problem, it has lower memory usage, so it's sometimes used for very large models for that reason. However, it doesn't save any evaluation time.

Optimizer	Seconds	AUC	Brier
GF	88s	0.802 ± 0.013	0.211 ± 0.008
GD	15s	0.785 ± 0.019	0.219 ± 0.009
SGD	15s	0.801 ± 0.012	0.215 ± 0.005
SGDM	27s	0.811 ± 0.020	0.214 ± 0.006
Signed Momentum	16s	0.819 ± 0.014	0.202 ± 0.007
Adam	16s	$\mathbf{0.839} \pm 0.006$	$\mathbf{0.196} \pm 0.003$

Table 9: Performance comparison of different optimization algorithms on the custom implementation of TCN. While slightly less effective than the off the shelf TCN, it's considerably faster, which makes it good for testing optimization ideas.

4.2.2 Adaptive Step Size / Schedules

In this section we will consider only Adam, and explore ways that the step size can be modified during training, known as a schedule or scheduler.

Warmup

It's well established that random initialization can produce atrocious conditioning. A simple solution is just to reduce the learning rate for a couple of epochs at first, in order to try to find a flatter, less difficult part of the loss landscape. This tends to prevent early divergence, and allow training at overall higher learning rates.

This can be thought of as related to cold-starting momentum estimates with zero, as the normal adam estimate corrects for this zero, and we're just putting it back.

This is very important with huge models, and very irregular data. In this case we've chosen one of the smaller models for demonstration, so it's not as important. It is a little more important to get the transformer model to train, although this is still very much a small model with few parameters and little training data.

Decay

For larger and more complex models, better results can also come from lowering the step size at the end of optimization. There are always various ideas about better ways to do this, but just as the torrent of papers "improving" on Adam have never yet replaced it, so Cosine decay remains the undisputed champion of off-the-shelf decay schedules. In practice, in training very large models, often the learning rate is simply changed by hand. (Zhang et al. 2022).

The exact value of warmup vs cosine appears to vary with a lot with small changes in the learning rate. At high learning rates, cosine outperforms warmup which outperforms constant learning rate. At lower learning rates, they're all much the same.

Prodigy

Another technique with a bit more track record is D-adaptation, which sets the learning rate based on the total distance traveled since the beginning. The logic is that if the recent gradient updates have not been canceling each other out, then it's okay to take larger step

sizes.

Of course this is a bit daft in a nonconvex problem where the conditioning changes as the problem evolves, and after a while we expect it to get worse. With extremely large momentum or giant batch sizes (both more common in the large model settings that are currently dominant in the literature), it might work better. Or maybe the trick is to clip gradients, so that there are no outlier updates. Regardless, this technique is well and properly useless on the transformer models in our setting.

The reason is that the D-Adaptation optimizer estimates the update size needed based on distance already traveled since initialization. When the model is working well, it slowly increases step size until it zeros in on the right scale to find a solution, then the gradient flattens out, and the optimizer starts making smaller updates (despite a constant learning rate). But, if the model begins to diverge, then the gradients steepen, the updates get bigger, and the learning rate continues to increase - a runaway reaction.

There's no real benefit over the one-cycle approach (L. N. Smith 2017), which at least runs a lot faster. In theory, at least you can just run it once, and it'll find a decent learning rate. In practice, that will diverge as mentioned above unless several of the hyperparameters are tuned, which is no better than just tuning learning rate directly.

Performance Comparison

For this section, we evaluate by testing performance of the Transformer model with the Adam optimizer in predicting next-step dynamics, and report the Root Mean Squared Error in this task. Linear warmup and cosine decay does outperform the other schedules, as expected. The performance of Prodigy is notably worse, and not worth the complications.

Scheduler	**RMSE**
Constant	1.616 ± 0.097
Warmup	1.637 ± 0.091
Warmup + Cosine	$\mathbf{1.540} \pm 0.056$
Prodigy	1.840 ± 0.050

Table 10: RMSE comparison from different schedulers, using Transformer and Adam. Lower is better

4.2.3 Gradient Modifications

The field of modifications to the gradient as an update is large, fast moving, and filled with irreproducible results. However, we test a couple of recent developments.

Schedule Free

There's been some recent interest around a tuning schedule which has been (optimistically) named "schedule free". The rough idea is to keep two locations: y, the current location, and x, a running arithmetic mean of y. The trick is that the update nudges y both away from x, and in the direction of zero from x. The term "schedule free" refers to the idea that no further variation in learning rate is necessary. (Defazio et al. 2024)

This is oddly effective when coupled with SAM. Because the two mechanisms of action are related, and the conceptual models offered by their creators don't really mesh together, it's quite hard to understand what the combination of the two is actually doing. Note that not only does it lower the RMSE when coupled with SAM and raise it when used without SAM, but the variance is substantially reduced when coupled with SAM, and increased when used without it.

Incidentally, it's not at all clear to me that the schedule free optimizer doesn't itself benefit from scheduling, but that seems out of scope to test.

SAM

This one is pretty straightforward. (Foret et al. 2021). The meta optimizer calls autograd once, then takes a step up instead of down the gradient. It then calls autograd a second time to get gradients at that second, worse point. Then it comes back down to the original point, and lets the base optimizer handle the gradients however it likes.

How do we reason about that? On a poorly conditioned, axis aligned parabola, the gradient in the direction of slow movement won't change at all. The gradient in the direction of fast movement will increase. This seems wild! It makes the conditioning of the problem effectively worse.

So this is probably some kind of edge of stability phenomenon, where SAM prevents momentum from smoothing out training as much, so higher momentum is usable, which in turn drives the model out of the sharpest basins, and that produces better generalization.

Weight Decay

Similar to L2 / Tikhonov / Ridge Regression, weight decay penalizes large parameter values, and will tend to shrink them back towards zero. This is popular outside of deep learning, with underparameterized models, but can also be used with overparameterized models.

Performance Comparison

Again, comparison is of RMSE in next-step dynamics prediction using a transformer and Adam. SAM produces a huge boost in performance, cutting RMSE by 30-50%. Weight decay seems to make a tiny difference across the board, but within the margin of error. Schedule Free only helps if used together with SAM, which I have oddly not seen reported in the literature.

Decay		**w/ SAM**	**w/o SAM**
w/o Decay	**w/ Schedule Free**	0.773 ± 0.012	1.666 ± 0.164
-	**w/o Schedule Free**	0.905 ± 0.032	1.581 ± 0.105
w/ Decay	**w/ Schedule Free**	$\mathbf{0.759} \pm 0.010$	-
-	**w/o Schedule Free**	0.854 ± 0.024	1.578 ± 0.061

Table 11: RMSE comparison for gradient modifications, using Transformer + Adam

Chapter 5

Results

The high level question this thesis seeks to explore is whether deep learning can process pen stroke data and classify Alzheimer's disease with reasonable accuracy. Using a sample of SDSTs administered to 83 patients, we find that performance is indeed promising. The AUC-ROC for classifying responses to individual questions (e.g. copying "12" into one box on the test) is not bad (about 85%). When the responses to all questions for a given individual are combined, we get an AUC of 97.25%, which is quite good.

Aggregation	Test-Set Size	AUC	SE
Single Response	10%	85.34%	0.01%
Whole Test	10%	97.25%	0.01%
Whole Test	40%	96.50%	0.1%

As a further check to avoid problems with small sample size, we retrain the model on only 60% of the data, and evaluate on fully 40% of the data, producing similar results.

5.1 Overall Architectural Approaches

Family	Model Type	AUC-ROC	Brier Score
Raster	2D CNN	0.680 ± 0.026	0.245 ± 0.007
Scalar	Logistic	0.723 ± 0.054	0.234 ± 0.019
1D	FFT	0.823 ± 0.024	0.213 ± 0.018
Causal	TCN	$\mathbf{0.855} \pm 0.005$	0.195 ± 0.003
Pretrained	LSTM	0.849 ± 0.009	$\mathbf{0.190} \pm 0.004$

Table 12: Model families and metrics from the best example. Standard deviation in parentheses, tuned on 10 runs.

Overall, models based on time and space data appear to outperform models based only on the space traversed by the pen. Narrowing in on what we tested more carefully, 1d sequence models substantially outperform logistic regression on scalar features. Causal CNNs appear to further outperform time-symmetric versions, although the difference is not as large. Pretraining the models by predicting future dynamics surprisingly does not improve classification performance, certainly not beyond the margin of error.

5.2 Feature ablations

added	first	second	last
Curvature	3.9%	-2.7%	0.9%
Time	4.5%	-1.1%	0.6%
$\frac{d\lvert\frac{dx}{dt}\rvert}{dt}$	8.3%	-1.0%	0.6%
$\frac{d^2x}{dt^2}$	12.4%	-0.0%	-0.2%
$\frac{d^3x}{dt^3}$	13.1%	1.5%	0.8%
$\frac{dx}{dt}$	15.1%	-	1.0%
all features	16%		
t, x, y	11.3%		

Table 13: Feature ablations. The standard error is about .1%. All of these take about 10 seconds to train on a macbook

Since features are correlated, it's not trivial to establish what is meant by the value added of a single feature. Overall, shapley values are an axiomatic approach that is well respected in the field, but require many model evaluations to calculate (Fryer, Strümke, and Nguyen 2021). In this case it would involve evaluating the model 640 times. Instead, we approximate the effect of features by studying

- the gain in quality of a model that uses only sequence length, if that feature is added in
- the loss of quality of a model with all features, if that feature is removed
- the gain in quality of a single feature model when a second feature is added

We find that velocity is the most important single feature, but that any 5 features produce an excellent model without the sixth. Since velocity is the most useful, we also

study the effect of adding one further feature, and find that although there is clearly some benefit from having them all, any one additional feature doesn't help much.

Retraining the model on raw spatiotemporal points reduces performance by about 5% of AUC. Presumably careful tuning could produce better results, but as it stands this is notably not as good as the engineered features.

5.3 Ablation of truncation

length	vs 128
32	-7.4%
64	-2.2%
128	0.0%
256	0.1%
512	-0.7%
1000	-1.8%

Table 14: Truncation ablation

Since the longest tail of sequences had to be cut out for the model to fit in memory, it's worth studying the effect of this truncation. We basically find that the model architecture and training parameters have been optimized for the sequence length for which it was tuned. It is likely that retuning could produce better models with longer sequences, though it is also possible that the additional noise would simply degrade performance.

Chapter 6

Discussion

After exploring various deep learning approaches, we found that temporal convolutional networks substantially outperformed both the results from a linear model, and from a visual model trained on the rasterized data. From an applied rather than methodological perspective, the theoretical motivations for deep learning are less important than the quality of the resultant model. The key ingredients we recommend to others are:

- Dilated, causal convolutional neural networks (TCN)
- The Adam optimizer
- Sharpness Aware Minimization

On a more procedural note, we highly recommend:

- Implementing early stopping based on validation performance, and then completely ignoring epoch-to-epoch results except during debugging.
- Using automated hyperparameter tuning tools. Even just basic Gaussian Process and Upper Confidence Bound based tools to prioritize and schedule experiments, and then good visualization tools makes a huge difference in mental load compared to manual testing, or grid search.
- Tuning hyperparameters for only a few tens of seconds to guess at a range of good values, and only then searching the range with many epoch-long runs. This makes a huge difference in the overhead of experiments.

- Keeping two holdout sets. After hyperparameter tuning each configuration on the validation set, we used the first holdout set to compare the best settings for different architectures, which provided much more confidence. We then saved the second holdout set for final evaluation once the work was done.

- Training on a GPU with CUDA. MacBook Metal Performance Shaders (MPS) still give something like 10x speedup over CPU, but there are innumerable places where pytorch tricks haven't been implemented for MPS.

6.1 Future Directions: Data

The data used in this study was collected in a relatively limited setting, and collecting more broadly may improve generalization, especially for underrepresented groups. In addition, our model is trained to predict the clinicians' contemporaneous judgment, but an even more powerful approach may be to collect and predict follow up data. If models can not only classify current state but predict future deterioration, this could be a major help to patients and their care providers in planning for the future.

6.2 Future Directions: Clinician Baselines

If we can work with clinicians to more closely understand the features they pay attention to in assessing the SDST, we may be able to not only give them the assessment, but try to identify cases where the assessments are counterintuitive.

6.3 Future Directions: Architecture

The surprisingly strong results from FFT suggest that we might be able to directly apply Fourier transformation at the level of individual strokes, not just responses. This might help address the immense (orders of magnitude) variability in stroke length and provide a better balance of performance on long and short strokes.

Stroke-level analysis may provide more useful lessons for handwriting analysis outside the SDST setting. It might also allow better attribution of predictions to the individual stroke level, which may be easier for clinicians to interpret.

A significant fraction of compute time appears to have been spent operating on padding used to make sequences the same length, so standardizing stroke length using FFT might also speed up training considerably.

Chapter 7

Conclusions

This thesis presents a novel deep learning approach for automated Alzheimer's diagnosis using handwritten responses to the Digit Symbol Substitution Test (DSST). By leveraging temporal convolutional networks trained on individual question responses and ensembling their predictions, we achieve state-of-the-art performance, with an AUC of 97.25% for patient-level classification. Careful architecture search revealed dilated convolutions, Adam optimization, and sharpness-aware minimization as crucial for this strong result.

The potential impact of this work is substantial. Alzheimer's disease devastates millions of patients and families worldwide, and treatments available or in the pipeline acan only slow rather than reverse deterioration. For millions who have access only to primary care doctors without specialized neuropsychological assessment training, and for whom seeking out specialized assessment can be frightening, the deployment of cheaper and less overwhelming early screening methods can be a game changer.

However, important work remains. While our results are promising, the models must be validated on larger and more diverse datasets, especially incorporating longitudinal outcomes. Investigating how our approach complements the strategies of expert clinicians may reveal further opportunities to assist their assessments. Methodologically, reducing the computational cost of training and inference will be crucial for scaling this approach to real-world applications.

Though the societal challenge of Alzheimer's is immense, this thesis demonstrates both the value of the Symbol Digit Substitution Test (SDST) as an evaluative tool, and the power of deep learning to extract clinically valuable insights from existing paper-based cognitive

tests. We are excited to apply and extend the academic and engineering lessons learned here to inform real world deployment in a practical setting.

Bibliography

Akaike, Hirotugu (1974). "A new look at the statistical model identification". In: *IEEE Transactions on Automatic Control* 19, pp. 716–723. URL: https://api.semanticscholar.org/CorpusID:411526.

Alzheimer's Association (2024). "2024 Alzheimer's disease facts and figures". In: *Alzheimer's & Dementia* 20.5, pp. 3708–3821. DOI: https://doi.org/10.1002/alz.13809. eprint: https://alz-journals.onlinelibrary.wiley.com/doi/pdf/10.1002/alz.13809. URL: https://alz-journals.onlinelibrary.wiley.com/doi/abs/10.1002/alz.13809.

Andersen, Stacy L. et al. (2021). "Digital Technology Differentiates Graphomotor and Information Processing Speed Patterns of Behavior". In: *Journal of Alzheimer's Disease* 82.1, pp. 17–32. DOI: 10.3233/JAD-201119.

Ayaz, Zainab et al. (2023). "Automated methods for diagnosis of Parkinson's disease and predicting severity level". In: *Neural Computing and Applications* 35.20, pp. 14499–14534. DOI: 10.1007/s00521-021-06626-y. URL: https://doi.org/10.1007/s00521-021-06626-y.

Belder, Christopher R S, Jonathan M Schott, and Nick C Fox (2024/07/15 2023). "Preparing for disease-modifying therapies in Alzheimer's disease". In: *The Lancet Neurology* 22.9, pp. 782–783. DOI: 10.1016/S1474-4422(23)00274-0. URL: https://doi.org/10.1016/S1474-4422(23)00274-0.

Belkin, Mikhail et al. (July 2019). "Reconciling modern machine-learning practice and the classical bias–variance trade-off". In: *Proceedings of the National Academy of Sciences* 116.32, pp. 15849–15854. ISSN: 1091-6490. DOI: 10.1073/pnas.1903070116. URL: http://dx.doi.org/10.1073/pnas.1903070116.

Campitelli, A. et al. (Jan. 2023). "A Novel Digital Digit-Symbol Substitution Test Measuring Processing Speed in Adults At Risk for Alzheimer Disease: Validation Study". In: *JMIR Aging* 6, e36663.

Chen, Liang-Chieh et al. (2018). "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40.4, pp. 834–848. DOI: 10.1109/TPAMI.2017.2699184.

Chung, Junyoung et al. (2014). *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling*. arXiv: 1412.3555 [cs.NE]. URL: https://arxiv.org/abs/1412.3555.

Cook, J. (2023). "An Effective Platform for Assessing Cognitive Health". MA thesis. MIT.

D'Alessandro, Tiziana et al. (2023). "A Machine Learning Approach to Analyze the Effects of Alzheimer's Disease on Handwriting Through Lognormal Features". In: *Graphonomics in Human Body Movement. Bridging Research and Practice from Motor Control to*

Handwriting Analysis and Recognition. Ed. by Antonio Parziale, Moises Diaz, and Filipe Melo. Cham: Springer Nature Switzerland, pp. 103–121. ISBN: 978-3-031-45461-5.

Dai, Andrew M and Quoc V Le (2015). "Semi-supervised Sequence Learning". In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes et al. Vol. 28. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2015/file/7137debd45ae4d0ab9aa953017286b20-Paper.pdf.

Dao, Tri et al. (June 2019). "Learning Fast Algorithms for Linear Transforms Using Butterfly Factorizations." eng. In: *Proc Mach Learn Res* 97, pp. 1517–1527. ISSN: 2640-3498 (Electronic).

Defazio, Aaron et al. (2024). *The Road Less Scheduled*. arXiv: 2405.15682 [cs.LG]. URL: https://arxiv.org/abs/2405.15682.

Demakis, G. J. et al. (Mar. 2001). "Incidental recall on WAIS-R digit symbol discriminates Alzheimer's and Parkinson's diseases". In: *J Clin Psychol* 57.3, pp. 387–394.

Deng, Li (2012). "The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]". In: *IEEE Signal Processing Magazine* 29.6, pp. 141–142. DOI: 10.1109/MSP.2012.2211477.

Fernandes, Carina Pereira et al. (2023). "Handwriting Changes in Alzheimer's Disease: A Systematic Review". In: *Journal of Alzheimer's Disease* 96.1, pp. 1–11. DOI: 10.3233/JAD-230438.

Foret, Pierre et al. (2021). *Sharpness-Aware Minimization for Efficiently Improving Generalization*. arXiv: 2010.01412 [cs.LG]. URL: https://arxiv.org/abs/2010.01412.

Fryer, Daniel, Inga Strümke, and Hien Nguyen (2021). "Shapley Values for Feature Selection: The Good, the Bad, and the Axioms". In: *IEEE Access* 9, pp. 144352–144360. DOI: 10.1109/ACCESS.2021.3119110.

Fu, Jingwen et al. (2023). *When and Why Momentum Accelerates SGD:An Empirical Study*. arXiv: 2306.09000 [cs.LG]. URL: https://arxiv.org/abs/2306.09000.

Genova, Lisa (2021). *Remember: The science of memory and the art of forgetting*. Harmony Books.

Gillies, Andrea (2011). *Keeper: One House, three generations, and a journey into alzheimer's*. Broadway Paperbacks.

Hochreiter, Sepp and Jürgen Schmidhuber (1997). "Long Short-Term Memory". In: *Neural Computation* 9.8, pp. 1735–1780.

Huang, L. (2017). "The digital symbol digit test : screening for Alzheimer's and Parkinson's". MA thesis. MIT.

Ioffe, Sergey and Christian Szegedy (2015). "Batch normalization: accelerating deep network training by reducing internal covariate shift". In: *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*. ICML'15. Lille, France: JMLR.org, pp. 448–456.

Jaeger, Judith (Oct. 2018). "Digit Symbol Substitution Test: The Case for Sensitivity Over Specificity in Neuropsychological Testing." eng. In: *J Clin Psychopharmacol* 38.5, pp. 513–519. ISSN: 1533-712X (Electronic); 0271-0749 (Print); 0271-0749 (Linking). DOI: 10.1097/JCP.0000000000000941.

Keuck, Lara (2018). "Diagnosing Alzheimer's disease in Kraepelin's clinic, 1909–1912". In: *History of the Human Sciences* 31.2, pp. 42–64. DOI: 10.1177/0952695118758879. eprint: https://doi.org/10.1177/0952695118758879. URL: https://doi.org/10.1177/0952695118758879.

Kingma, Diederik P. and Jimmy Ba (2017). *Adam: A Method for Stochastic Optimization*. arXiv: 1412.6980 [cs.LG]. URL: https://arxiv.org/abs/1412.6980.

Knopman, David S. and Linda Hershey (Oct. 2023). "Implications of the Approval of Lecanemab for Alzheimer Disease Patient Care". In: *Neurology* 101.14, pp. 610–620. ISSN: 1526-632X. DOI: 10.1212/wnl.0000000000207438. URL: http://dx.doi.org/10.1212/WNL.0000000000207438.

Kumar A Sidhu J, Goyal A et al. (2022). *Alzheimer Disease*. Treasure Island (FL): StatPearls Publishing.

Lea, Colin et al. (2016). "Temporal Convolutional Networks: A Unified Approach to Action Segmentation". In: *Computer Vision – ECCV 2016 Workshops*. Ed. by Gang Hua and Hervé Jégou. Cham: Springer International Publishing, pp. 47–54. ISBN: 978-3-319-49409-8.

LeCun, Y. et al. (Dec. 1989). "Backpropagation Applied to Handwritten Zip Code Recognition". In: *Neural Computation* 1.4, pp. 541–551. ISSN: 0899-7667. DOI: 10.1162/neco.1989.1.4.541.

Lee-Thorp, James et al. (2021). "Fnet: Mixing tokens with fourier transforms". In: *arXiv preprint arXiv:2105.03824*.

Lesoil, C. et al. (Apr. 2023). "Validation study of 'Santé-Cerveau', a digital tool for early cognitive changes identification". In: *Alzheimers Res Ther* 15.1, p. 70.

Lezak, Muriel Deutsch (2004). *Neuropsychological assessment*. Oxford University Press, USA.

Maurer, Konrad, Stephan Volk, and Hector Gerbaldo (1997). "Auguste D and Alzheimer's disease". eng. In: *The Lancet (British edition)* 349.9064, pp. 1546–1549. ISSN: 0140-6736.

McKeown, Alex et al. (2020). "Health Outcome Prioritization in Alzheimer's Disease: Understanding the Ethical Landscape." eng. In: *J Alzheimers Dis* 77.1, pp. 339–353. ISSN: 1875-8908 (Electronic); 1387-2877 (Print); 1387-2877 (Linking). DOI: 10.3233/JAD-191300.

Moetesum, Momina et al. (2022). "A survey of visual and procedural handwriting analysis for neuropsychological assessment". In: *Neural Computing and Applications* 34.12, pp. 9561–9578. DOI: 10.1007/s00521-022-07185-6. URL: https://doi.org/10.1007/s00521-022-07185-6.

Mucha, Jan et al. (2023). "Exploration of Various Fractional Order Derivatives in Parkinson's Disease Dysgraphia Analysis". In: *ArXiv* abs/2301.08529. URL: https://api.semanticscholar.org/CorpusID:254928492.

Nandi, Arindam et al. (2024). "Cost of care for Alzheimer's disease and related dementias in the United States: 2016 to 2060". In: *npj Aging* 10.1, p. 13. DOI: 10.1038/s41514-024-00136-6. URL: https://doi.org/10.1038/s41514-024-00136-6.

Nolazco-Flores, Juan Arturo et al. (2021). "Exploiting spectral and cepstral handwriting features on diagnosing Parkinson's disease". In: *IEEE Access* PP, pp. 1–1. URL: https://api.semanticscholar.org/CorpusID:240004346.

Patnode CD Perdue LA, Rossom RC et al. (2019). *Screening for Cognitive Impairment in Older Adults: An Evidence Update for the U.S. Preventive Services Task Force*. Tech. rep. 189. Rockville, MD: Agency for Healthcare Research and Quality.

Prange, Alexander and Daniel Sonntag (2022). "Modeling users' cognitive performance using digital pen features". In: *Frontiers in Artificial Intelligence* 5, p. 58.

Press, Ofir, Noah A. Smith, and Mike Lewis (2022). *Train Short, Test Long: Attention with Linear Biases Enables Input Length Extrapolation*. arXiv: 2108.12409 [cs.CL]. URL: https://arxiv.org/abs/2108.12409.

Radford, Alec et al. (2019). "Language Models are Unsupervised Multitask Learners". In: *OpenAI Blog*.

Rahimi, Ali and Benjamin Recht (2007). "Random Features for Large-Scale Kernel Machines". In: *Advances in Neural Information Processing Systems*. Ed. by J. Platt et al. Vol. 20. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2007/file/013a006f03dbc5392effeb8f18fda755-Paper.pdf.

Ramachandran, Prajit, Barret Zoph, and Quoc V. Le (2017). *Searching for Activation Functions*. arXiv: 1710.05941 [cs.NE]. URL: https://arxiv.org/abs/1710.05941.

Richardson, L. F. (1911). "The Approximate Arithmetical Solution by Finite Differences of Physical Problems Involving Differential Equations, with an Application to the Stresses in a Masonry Dam". In: *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character* 210, pp. 307–357. ISSN: 02643952. URL: http://www.jstor.org/stable/90994 (visited on 08/04/2024).

Smith, Leslie N. (2017). *Cyclical Learning Rates for Training Neural Networks*. arXiv: 1506.01186 [cs.CV]. URL: https://arxiv.org/abs/1506.01186.

Vaswani, Ashish et al. (2017). "Attention is All you Need". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845Paper.pdf.

Vollset, Stein Emil et al (2024/07/14 2024). "Burden of disease scenarios for 204 countries and territories, 2022–2050: a forecasting analysis for the Global Burden of Disease Study 2021". In: *The Lancet* 403.10440, pp. 2204–2256. DOI: 10.1016/S0140-6736(24)00685-8. URL: https://doi.org/10.1016/S0140-6736(24)00685-8.

Wengert, R. E. (1964). "A simple automatic derivative evaluation program". In: *Communications of the ACM* 7, pp. 463–464. URL: https://api.semanticscholar.org/CorpusID:24039274.

Werner, Perla et al. (2006). "Handwriting process variables discriminating mild Alzheimer's disease and mild cognitive impairment." In: *The journals of gerontology. Series B, Psychological sciences and social sciences* 61 4, P228–36. URL: https://api.semanticscholar.org/CorpusID:13882317.

Williamson, M. et al. (Dec. 2022). "Validation of a digit symbol substitution test for use in supervised and unsupervised assessment in mild Alzheimer's disease". In: *J Clin Exp Neuropsychol* 44.10, pp. 768–779.

Zhang, Susan et al. (2022). *OPT: Open Pre-trained Transformer Language Models*. arXiv: 2205.01068 [cs.CL]. URL: https://github.com/facebookresearch/metaseq/blob/main/projects/OPT/chronicles/10_percent_update.md.