Robust Gait Recognition Techniques for Wearable Devices

Jon Nelson

Submitted in partial fulfillment of the requirements of the degree of Bachelor of Science with Honors in the School of Engineering at Brown University.

Prepared under the direction of

Prof. Sherief Reda, Advisor

Prof. Jacob Rosenstein, Reader

_____

Advisor's Signature

_____

Reader's Signature

_____

Honors Chair's Signature

Computer Engineering
Brown University
April 28, 2020

Contents

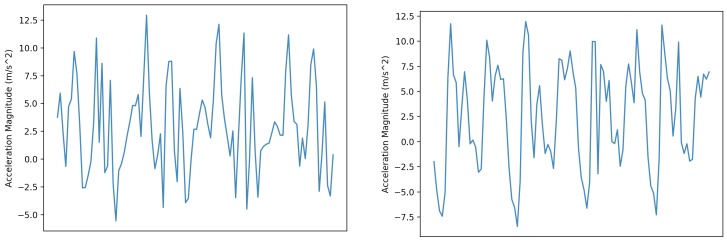
1 Introduction

With the widespread use of motion sensors in embedded devices such as phones and smart watches, there is a vast amount of data available for analysis. Motion sensor data from these devices can be leveraged to provide a wide-range of technologies such as health monitoring and activity tracking [8] [9]. However, in order for these applications to be robust in real-world settings, we must account for sensor inconsistencies that can significantly distort the data. For instance, simply rotating an accelerometer sensor will vastly change its measurements. We focus on applications toward gait recognition and discuss techniques for handling variability in sensor data. We also propose a novel feature extraction method for time-series data that leverages fractional order derivatives.

1.1 Gait Recognition

Gait recognition is a form of biometric security that attempts to identify a user by their unique walking pattern. There are many different proposed methods for capturing gait measurements ranging from video analysis to floor sensors [10] [11] [31]. Wearable motion sensors have gained growing interest for use in gait recognition due to their ability to capture unique walking signatures [14]. As shown in Figure 1, the shape of each person's accelerometer signal during walking is identifiably different. Although our study focuses on accelerometers, gyroscopes can also be used with similar effectiveness.

Conveniently, accelerometers can be found in almost all smart-phones. Similar to the use of facial recognition for protection of smart-phones, gait recognition is a reliable method for verifying that your phone is in the right hands as you walk around. Current research has shown promising results in this field when data is collected in a controlled environment, but there is a significant drop-off



(a) Subject 1 (b) Subject 2

Figure 1: Roughly seven steps of accelerometer magnitude data for two different subjects

in performance when accounting for realistic variables such as arbitrary sensor orientation and placement [1] [5] [6] [7].

2 Related Work

The main challenge when analyzing accelerometer data is the issue of orientation. Accelerometers measure the direction and magnitude of acceleration. However, the direction of acceleration is with respect to the axes of the sensor itself (see Figure 2) and so rotating the sensor will also rotate your acceleration data. This presents issues if your analysis depends on the raw three-dimensional signal.

There have been many proposed approaches to calculate rotation-invariant features from three-dimensional acceleration data. The simplest approach is to analyze only the acceleration magnitude. This sacrifices potentially useful information about the direction of acceleration. However, we can compensate for this information loss by transforming our signal into rich 2D feature representations using techniques such as the Short-time Fourier transform and the continuous wavelet transform [18] [22] [23] [24] [25] [26] [27].

A more creative rotation-invariant method, proposed by Zhong *et al.*, is to

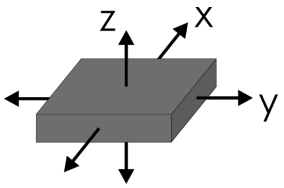


Figure 2: Axes of accelerometer. Figure taken from [17].

use the angles between each pair of acceleration vectors in the signal [1]. As long as the orientation is consistent for all pairs of samples, the angle between the vectors is invariant to rotation of the sensor. Similarly, Kobayashi *et al.* use the autocorrelation matrix of the signal in the frequency domain as features for their analysis [2]. Since the autocorrelation matrix of a three-dimensional signal is estimated using the inner product of each possible pair of vectors in the signal, this method indirectly leverages the rotation-invariant property of inner products.

Another strategy is to represent our acceleration signal along components that are independent of the sensor's axes. For instance, Gadaleta *et al.* and Mantyjarvi *et al.* use Principal Component Analysis to determine the direction of greatest variance and then transform the acceleration signal along this axis [3] [19]. We can also use Independent Component Analysis to statistically determine the axes along which our data is maximally independent. Since the statistical properties of variance and independence are determined by the nature of the user's movement and not by the sensor, PCA and ICA are invariant to sensor rotation.

Finally, Subramanian *et al.* attempt to correct for rotation factors by applying the Kabsch algorithm on each pair of acceleration signals to estimate their relative orientations before computing their similarity. [4].

In addition to rotation-invariant feature extraction techniques, there are many other steps in the analysis flow that contribute to handling inconsistent

sensor data. This thesis will thoroughly address each stage of the machine learning pipeline starting with data preprocessing, then feature extraction and finally classification.

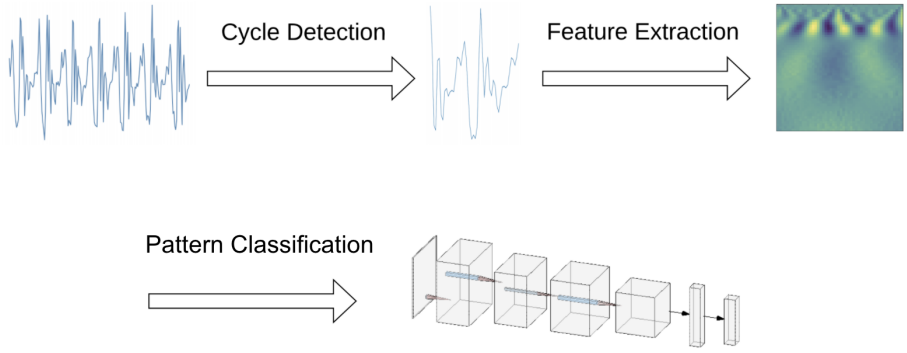


Figure 3: Gait Recognition Pipeline

3 Data Preprocessing

Accelerometers are highly prone to noise and so it is important to first extract meaningful signals before performing analysis. While the overall walking signature is crucial for gait recognition, each twitch and micro-movement of the sensor will also factor into the readings. Accelerometers operate by measuring the inertial force of a suspended mass. The accumulation of small vibrations of this mass can cause high levels of noise. Denoising techniques such as averaging are often implemented in the hardware of the accelerometer itself, but additional inconsistencies arise when data is collected by a multitasking operating system such as a smart device.

3.1 Resampling

In smart devices it is impossible to retrieve measurements at a fixed frequency since the processor is often running many different programs simultaneously and cannot guarantee the exact timing that it executes its tasks. The standard deviation of the accelerometer sampling frequency can be as high as 5Hz for a typical smart-phone. When analyzing data from a multi-tasking operating system, the first step is to resample the measurements to a fixed sampling frequency using the timestamps of each sample. This will avoid temporal distortions of our walking signal that occur if we assume that samples are equally spaced.

It is important to note that we can avoid the computational overhead of resampling by using an embedded device with a real-time operating system. This is ideal for wearable sensor applications since these devices can guarantee precise sample timing. However, multi-purpose smart-devices are more widely applicable and so we cannot rely on fixed sampling rates.

3.1.1 Cubic Spline Interpolation

One strategy for resampling is to approximate the walking signal as a piecewise function of cubic polynomials [5]. We assign a cubic polynomial for each interval between accelerometer samples with the condition that adjacent polynomials are continuous at the point of overlap up to the second order derivative. This can be formalized as follows:

Given a set of $n+1$ data points x_i, y_i for $i = 0, \ldots, n$ where x_i represents the ith timestamp and y_i represents the ith acceleration reading, we can fit the following piecewise function to our data:

$$S(x) = \begin{cases} \mathrm{C}_1(x) & \text{for } x_0 \leq x \leq x_1 \\ \vdots & \\ \mathrm{C}_n(x) & \text{for } x_{n-1} \leq x \leq x_n \end{cases}$$

where $C_i(x)$ is a cubic polynomial such that

$$C_i(x_i) = C_{i+1}(x_i)$$
$$C_i'(x_i) = C_{i+1}'(x_i)$$
$$C_i''(x_i) = C_{i+1}''(x_i)$$

Since each cubic polynomial has four coefficients we must solve for $4n$ variables. The boundary conditions for each polynomial gives us $4n$ constraints, allowing us to have a closed form solution to the system of equations.

It can be proven that cubic spline interpolation minimizes bending while fitting all data points [13]. Specifically, it minimizes the energy integral $\int (\frac{d^2 S(x)}{dx^2} dx)^2$ over all functions $S(x)$ that fit the data. Minimal bending is desirable because this is intuitively the smoothest fitting of our data.

Once we have approximated our data points as a piecewise polynomial function we can now evaluate our function at equally spaced time intervals in order to resample to a constant frequency. Figure 4 illustrates how resampling can fix temporal distortions that arise from a variable sampling frequency.

3.2 Filtering

Accelerometer data has a significant amount of high-frequency noise due to its sensitivity. Since we are only interested in capturing the overall gait dynamics, which are relatively low-frequency, we can apply a low-pass filter to reduce noise in our accelerometer data. A standard low-pass filter is the Butterworth filter which has a flat frequency response in the pass-band and can be tuned to sharply

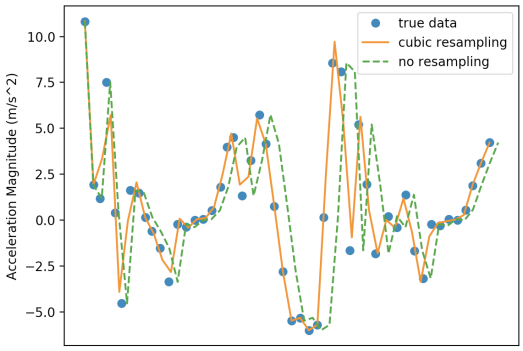


Figure 4: Resampling using Cubic Spline Interpolation to avoid temporal drift

cut-off high-frequency components [21]. This proves to be an effective method for isolating our walking signal from the accelerometer noise. As you can see in Figure 5, the Butterworth filter smooths out jagged edges of our raw acceleration signal to create a waveform that better represents the overall movement. The cut-off frequency and order of the filter were chosen in order to denoise without eliminating distinguishing characteristics of the signal.

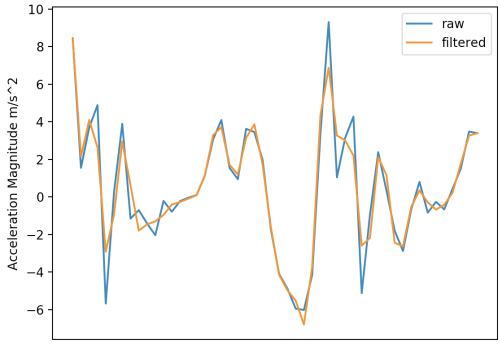
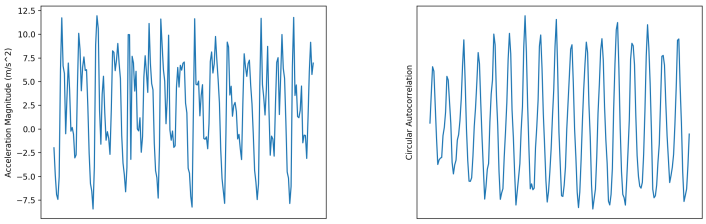


Figure 5: Effect of filtering on two seconds of acceleration magnitude data using order 3 Butterworth low-pass filter with cut-off frequency of 8Hz



(a) Raw Magnitude Signal (b) Circular Autocorrelation

Figure 6: Estimating the period of the raw acceleration signal using its circular autocorrelation

3.3 Cycle Detection

The final preprocessing step is to segment our walking data into gait cycles. This is necessary to ensure that our signal is consistently aligned. We break up our walking signal into two-second segments each starting at the beginning of a step. Our step detection algorithm first estimates the period of the gait cycle (the amount of time to take two steps) and then finds the strongest peaks that are approximately one period apart from each other.

3.3.1 Period Estimation

To estimate the period of the gait cycles we first calculate the circular autocorrelation of the signal, which is the correlation of the signal with itself for various time delays [5]. Note that these time-delays are applied circularly so that the end of the time-delayed signal is shifted onto the beginning of the original. There will be high values of correlation when the signal is delayed by a multiple of its approximate period since this will result in maximal overlap. By analyzing the time between peaks in the circular autocorrelation signal we can easily determine the period. Figure 6 shows how this method reveals a clean representation of our walking signal's periodicity.

The Weiner-Khinchin Theorem presents a simple method for calculating the circular autocorrelation $R(\tau)$ for all possible time delays τ by multiplying the Fourier transform of the signal with its conjugate and then transforming back to the time domain [12]:

$$R(\tau) = \mathcal{F}^{-1}(\mathcal{F}(x) \cdot \mathcal{F}^{*}(x))$$

Each peak in the autocorrelation signal corresponds with a step and so we estimate the period of a complete walking cycle as twice the average distance between these peaks.

3.3.2 Peak Detection

Once we have estimated the period, finding the beginning of each gait cycle is straight-forward. We simply find the strongest peak that is within a margin of 20% of the estimated period. We evaluate the strength of each peak as the relative height distance between its neighbors. Figure 7 shows how our peak detection algorithm segments a raw walking signal into gait cycles.

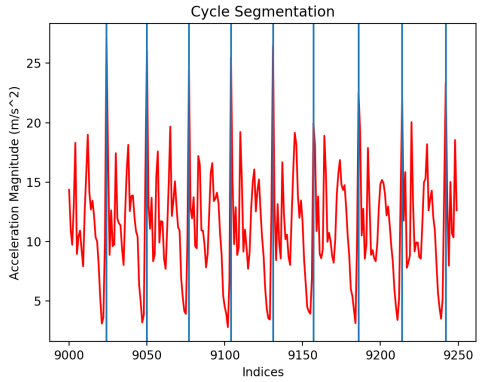


Figure 7: Segmentation of raw walking data into individual gait cycles

4 Feature Extraction

The next stage in our gait recognition pipeline is to extract meaningful characteristics from our walking signal. For simplicity's sake, our figures so far have focused on displaying the acceleration magnitude; however, our walking signal is three dimensional with an x, y, and z component that each contribute information about the gait dynamics. By extracting the distinguishing characteristics of these signals we can provide our classification model with more refined features than the raw signal itself.

4.1 Statistical Features

A computationally efficient approach is to calculate simple statistical measures of the signal. This includes calculating the mean, standard deviation, kurtosis, range, energy, bandpower, etc. [14]. Each of these features are often calculated for each dimension of the signal. This method sacrifices fine-grained information about the walking signal for broad statistical summaries, which can work well for classification problems with relatively few classes, but may not be sufficiently complex for harder problems. In addition, the statistical features for the individual x, y, z signals of the acceleration data are vulnerable to arbitrary rotations of the sensor, which can lead to poor performance in realistic settings.

4.2 Rotation-Invariant Techniques

To maximize the amount of information in our feature representation, we hope to leverage the direction of acceleration in addition to its magnitude. Unfortunately, we cannot rely on a fixed orientation of the sensor because this will fail in settings where the sensor is placed imperfectly. This seems like an impossible challenge since the direction of acceleration is entirely dependent on the orientation of the sensor. However, if we assume that the rate of the sensor's rotation

is slow, such that the orientation for consecutive readings is consistent, we can still produce powerful features beyond just analysis of the magnitude signal. In this section, we discuss various techniques that attempt to preserve some aspect of the three-dimensional signal while maintaining rotation-invariance.

4.2.1 Inner Products

An insight made by Zhong *et al.* is that we can use the inner product between acceleration vectors as a rotation-invariant feature [1]. The inner product can be simplified as a measure of the angle between two vectors scaled by their magnitudes. Intuitively, we note that no matter how you rotate this pair of vectors, the angle between them will remain the same. In the special case where the inner product is taken with respect to the vector and itself, we get the vector's magnitude. This rotation-invariant property of the inner product is shown more formally below with the unitary matrix R representing an arbitrary rotation on two accelerometer vectors a_1 and a_2.

$$\langle Ra_1, Ra_2 \rangle = a_1^T R^T R a_2 = a_1^T a_2 = \langle a_1, a_2 \rangle$$

We can leverage this property of inner products to create 2D feature representations of our signal that are rotation-invariant. By taking the outer product of the three dimensional acceleration signal with itself, we can produce a matrix of inner products between each possible pair of acceleration vectors. This matrix is an estimate of the autocorrelation matrix of the acceleration signal and can be used as our feature image. This technique can also be applied in the frequency domain with similar effectiveness [2]. Taking this idea further, we can even use the inner products between gradient vectors of the acceleration signal, which has shown to be a rich feature representation as well [5].

4.2.2 Principal Component Analysis

Principal Component Analysis is a statistical method for determining the directions of greatest variance in our data. Since the direction of variance in our acceleration data is completely determined by the user's movement, it is invariant to the orientation of the sensor [3]. More formally, given a $3 \times n$ matrix X of acceleration data where n is the number of samples, we can perform PCA by singular value decomposition of X into $U\Sigma V^T$ where U and V are unitary matrices and Σ is a diagonal matrix. $U\Sigma$ is our desired projection of X along the principal components (i.e. the right singular vectors). Since the singular value decomposition of X is equivalent to the eigenvalue decomposition of $X^T X$, it is rotation invariant (let $Y = RX$ for an arbitrary rotation matrix R acting on our acceleration data X, we have $Y^T Y = X^T R^T RX = X^T X$).

Once we have rotated our acceleration data along its principal components we can use any method to extract features from our data and our feature vector will remain rotation-invariant [19]. However, PCA is a statistical technique and since our walking data is limited to approximately two seconds per segment it is difficult to accurately find the true principal components.

4.2.3 Independent Component Analysis

Independent Component Analysis is another statistically-driven technique for finding interesting directions in our data [20]. ICA attempts to find a rotation of the data along components that are maximally independent from each other. Independence can be measured by the non-Gaussianity of the component, which is approximated by the kurtosis [15]. Applying ICA to our acceleration data allows us to orient our data along statistically promising directions rather than along the sensor's axes. Unfortunately, ICA also introduces additional complications such as scaling and random permutations of the component ordering.

Mantyjarvi *et al.* attempts to resolve this issue by ordering components based on their kurtosis [19]. However, ICA remains vulnerable to the same challenges as other statistical methods since it relies on larger acceleration signals for its estimations.

4.3 2D Feature Representations

We can dramatically increase the expressiveness of our feature representation by transforming our one-dimensional time-series waveforms into two-dimensional feature maps. For instance, we can represent our signal in both the time and frequency domain by performing either the Short-time Fourier transform or the continuous wavelet transform [18] [28]. This gives us a much richer representation of our signal and allows us to use more powerful classification methods. Analogous to tracking the frequency spectrum of our signal in time, we can also construct an image representing the change in shape of our signal over time using fractional order derivatives.

4.3.1 Short-time Fourier Transform

The Short-time Fourier transform is an industry standard technique for displaying the relationship between the time and frequency domain of a signal [28]. The hope is that through taking the Fourier transform of small intervals of the signal, we will be able to determine the frequency spectrum for local points in time. In reality, we can only calculate a small range of frequencies if our time interval is small. This is a significant issue when our signal is already short to begin with (in our case two seconds). On the other hand, if we instead take the Fourier transform of larger intervals of our signal, we then begin to lose locality in time. This trade-off is unavoidable as it is impossible to have perfect precision in both the frequency and time domain.

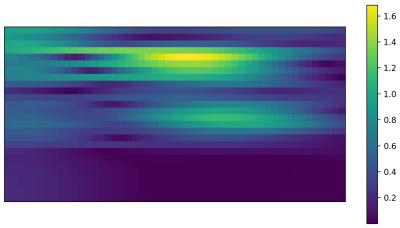


Figure 8: Spectrogram of two-second magnitude signal calculated using Short-time Fourier Transform

4.3.2 Continuous Wavelet Transform

The wavelet transform proves to be a much more effective method for generating time-frequency mappings for short waveforms since it is better equipped for producing local measurements of frequency. It has been well-studied as a form of feature representation and has shown to be an effective tool for gait analysis [18] [22] [23] [24] [25] [27] [28]. The main difference is that while the Fourier transform represents the time-domain signal as a linear combination of non-local sinusoids, the wavelet transform instead uses localized wavelets to analyze the signal.

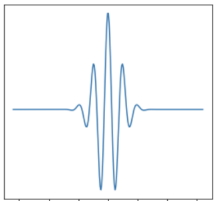


Figure 9: Morlet wavelet

Larger scalings of this wavelet represent smaller frequencies and so by convolving the time-domain signal with a range of scalings of the wavelet we can create an image of the signal's frequency spectrum with respect to time. This is often referred to as a scaleogram given that it is technically a function of the

scaling of the wavelet rather than of frequency. Most of the temporal information content of the wavelet transform is concentrated in low scalings of the wavelet (high end of the frequency spectrum) as these are the most local. As the scaling increases, we once again sacrifice locality for frequency resolution.

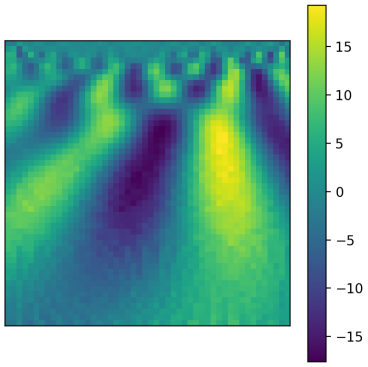


Figure 10: Scaleogram of two-second magnitude signal calculated using the continuous wavelet transform

4.3.3 Fractional Order Derivative Transform

While the previous methods are well-known, we also propose a novel method for feature mapping that expresses equally rich information content. Extending the notion of frequency in the previous methods to simply the rate of change, we can construct an image that displays the spectrum of derivatives of our signal using fractional order calculus. Specifically, we construct our image by taking fractional order derivatives of our time-series signal at discrete increments ranging from order zero to some fixed maximum order.

Integer order derivatives express valuable information about our signal such as the rate of change and curvature at instantaneous points in time. By filling in the gaps with fractional order derivatives, we can further refine our feature

representation to contain non-local characteristics as well.

We can decompose fractional derivatives into two steps: integer order differentation and fractional integration as shown below where α is the fractional order and n is an integer.

$$\mathcal{D}^{\alpha}(x) = J^{n-\alpha} D^{n}(x)$$

We define fractional order integration by using the Cauchy formula for repeated integrals where n is the number of consecutive integrations [16]:

$$J^{n}(x) = \frac{1}{(n-1)!} \int_{0}^{x} (x-t)^{n-1} f(t) dt$$

Approximating the factorial with the gamma function we have our fractional order integral:

$$J^{\alpha}(x) = \frac{1}{\Gamma(\alpha)} \int_{0}^{x} (x-t)^{\alpha-1} f(t) dt$$

Since fractional order derivatives are defined using integration over our signal, they represent non-local information. This allows them to capture more general characteristics about the overall waveform than integer order derivatives which represent instantaneous changes. Notice that this combination of local and non-local temporal information is also a strength of the wavelet transform. Unfortunately, fractional order derivatives are computationally expensive to compute as we must compute an integral for each point in our signal.

The algorithm to transform the time-series waveform into a 2-D fractional order derivative image is formally defined in Algorithm 1.

Algorithm 1: Fractional Order Derivative Image

Data: a: array holding acceleration signal of length n, $minorder$: minimum order of fractional derivative range, $maxorder$: maximum order of fractional derivative range, m: number of fractional order derivatives to calculate

Result: $fodimage$: $m \times n$ matrix representing the desired feature map

$i = 0$

for α *in linspace(minorder, maxorder, m)* **do**

  $fodimage[i, :] = \mathcal{D}^{\alpha}(a)$

  $i = i + 1$

end

return *fodimage*

5 Classification

Depending on our choice of data representation, we have various techniques for classification. Given a vector of statistical features we can simply use a fully connected neural network or any other black box machine learning classifier such as Support Vector Machines, k-Nearest-Neighbors, Decision Trees, etc. If we instead choose to use a rotation-invariant signal such as the magnitude, vector angles, PCA or ICA, then we can use a 1D convolutional neural network to classify our waveform. Finally, we can transform our raw waveform into a 2D feature image and use 2D convolutional neural networks to analyze the images.

5.1 Convolutional Neural Networks

Convolutional neural networks detect meaningful features by convolving different weight matrices over the input data. These weight matrices are called kernels and are optimally tuned during training to detect the most relevant features for classification. The convolutional layers then feed into a final subnetwork of fully connected layers that perform the classification.

Rather than calculating statistical features by hand, we instead allow our

network to find the optimal features in our data that maximize classification performance. This proves to be a much more effective method than hand-crafting our feature vector using heuristics.

However, given the high complexity level of these networks, we must be careful not to overfit our training data. Fortunately, there are many techniques for managing overfitting in CNN's including the use of pooling and dropout layers. Pooling layers summarize the learned features by grouping them into small subgroups and using only the maximum feature in the subgroup. This allows us to simplify our feature map to make it more manageable. We can also use dropout layers to regulate our network's complexity. Dropout layers randomly zero-out a fraction of the layer weights. This again reduces the complexity of our network and forces all layer weights to meaningfully contribute to the network. Our particular chosen neural network architectures for the 1D and 2D cases are illustrated in Figures 11 and 12.

6 Experimental Results

In order to evaluate the performance of the previously described methods, we applied each technique to the problem of gait recognition. In particularly, we focused on performance in worst-case scenarios where we must contend with measurement inconsistencies.

6.1 McGill Dataset

We used the McGill Dataset to test our methods because this is the closest approximation to real-world use cases. In this dataset, walking data is collected for twenty different subjects on two separate days. On each day approximately fifteen minutes of accelerometer readings are recorded. Between days, subjects changed their clothes and shoewear. In addition, there is no guarantee that

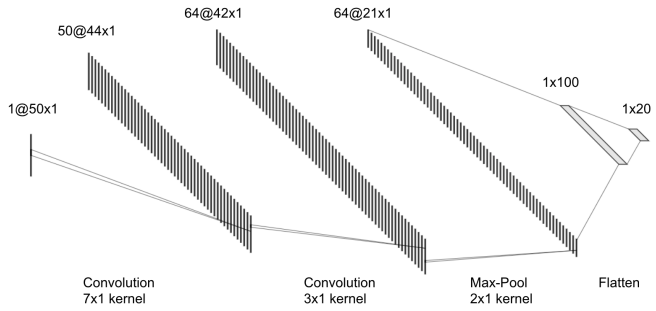


Figure 11: Architecture for 1D convolutional neural network (not shown is 50% dropout layer after second convolutional layer). Figure was generated using a software tool by Alex Lenail available at http://alexlenail.me/NN-SVG/LeNet.html.

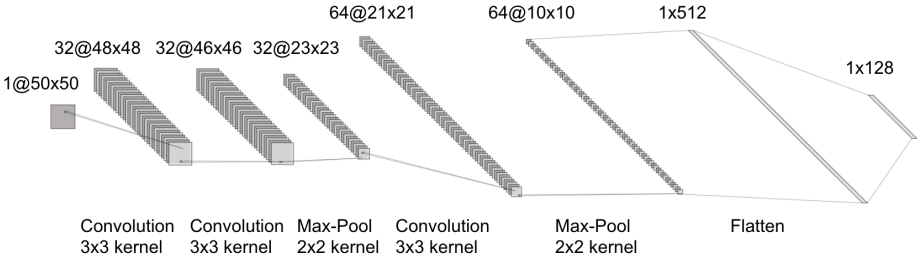


Figure 12: Architecture for 2D convolutional neural network (not shown is 25% dropout layer after each convolutional and fully connected layer except for the first). Figure was generated using a software tool by Alex Lenail available at http://alexlenail.me/NN-SVG/LeNet.html.

the sensor remained in the same orientation across days [7]. This dataset is available at the following link: https://www.cs.mcgill.ca/ jfrank8/data/gait-dataset.html.

6.2 Experimental Setup

We trained each model on 80% of the Day 1 walking data and tested on both 20% of the Day 1 data and 100% of the Day 2 data. During preprocessing, we segmented the data into two-second walking signals as this is enough for approximately two step cycles. We tested our methods on the harder task of classification rather than authentication. The distinction is that in classification our model must identify which person the walking signal belongs to rather than whether it belongs to a particular person or not.

6.3 Results

We first show the effect of each preprocessing step on the overall performance of our pipeline. Table 1 demonstrates that each additional step of preprocessing contributes to the Day 2 performance. In each case, we evaluate the Day 1 and Day 2 test accuracy using the magnitude signal as our input vector to a 1d convolutional neural network. This feature extraction and classification method can be thought of as a baseline for rotation-invariant techniques since the magnitude signal is the simplest such method. Note that each preprocessing step is performed in conjunction with all steps in prior rows of the table. For example, the bottom row represents the performance of using resampling, walking detection and filtering all together.

It is important to note that the McGill dataset contains a significant amount of non-walking data, which must be parsed out using a walking detection algorithm. In our case we used Frank *et al.*'s suggestion of simply evaluating the

mean of the absolute value of the magnitude for each segment and treating all data below a certain threshold as non-walking data.

Method	Day 1 Test Accuracy (%)	Day 2 Test Accuracy (%)
Nothing	89.38	42.48
+ Resampling	84.83	46.11
+ Walking detection	98.01	55.28
+ Filtering	93.85	59.69

Table 1: Effect of adding various preprocessing steps

Finally, we evaluate each of the previously discussed feature extraction and classification methods on the McGill Dataset. Note that in each case we use consistent preprocessing steps of resampling, walking detection, filtering, and cycle segmenting. We report the test accuracy for Day 1 and Day 2 for each method in Table 2. A detailed description of each method is presented below Table 2 for reference. Some of these techniques are well-known such as Principal Component Analaysis (PCA), the Short-time Fourier transform (STFT), and the continuous wavelet transform (CWT). Others are experimental methods recently proposed by gait analysis researchers such as the inner product methods (implemented here by calculating the autocorrelation matrix of either the time or frequency domain signals) [1] [2] [5]. Finally, the fractional order derivative imaging technique is our own novel contribution to this field.

Method	Day 1 Test Accuracy (%)	Day 2 Test Accuracy (%)
CWT	96.51	67.02
FODI (our approach)	96.03	66.57
Time-domain Autocorr [1]	98.01	65.28
Freq-domain Autocorr [2]	98.15	65.22
HOS [6]	—	64.50
STFT	95.62	59.77
Magnitude	93.85	59.69
PCA [19]	95.42	48.18
ICA [19]	91.42	47.89
TDEBOOST [7]	—	42.40
Baseline	99.52	27.55
Statistical features	99.56	24.75

Table 2: Comparing Performance on McGill Dataset

CWT: We take the continuous wavelet transform of the acceleration magnitude signal using Haar wavelets to produce a scaleogram. We then use a 2D convolutional neural network to perform image classification on the scaleogram.

FODI: We take the fractional order derivative image of the acceleration magnitude signal using the method described in Algorithm 1. We then use a 2D convolutional neural network to perform image classification.

Time-domain Autocorr: We take the outer product of the three-dimensional time domain signal with itself to create a 2D image. This leverages the rotation invariance of inner products which was first suggested by Zhong *et al.* [1]. We then use a 2d convolutional neural network for image classification.

Freq-domain Autocorr: We take the autocorrelation of the three-dimensional Fourier transform signal to create a 2D image. We use a 2d convolutional neural network for image classification. This method was proposed by Kobayashi *et al.* [2].

HOF: We report the published results of Sprager *et al.* [6]

STFT: We perform the Short-time Fourier transform of the acceleration magnitude signal to create a 2d spectrogram. We use a 2d convolutional neural network for image classification.

Magnitude: We use a 1d convolutional neural network on the acceleration magnitude signal.

PCA: We perform Principal Component Analysis to rotate the three-dimensional acceleration signal onto its principal components [19]. We then use a 1d convolutional neural network on the rotated signal.

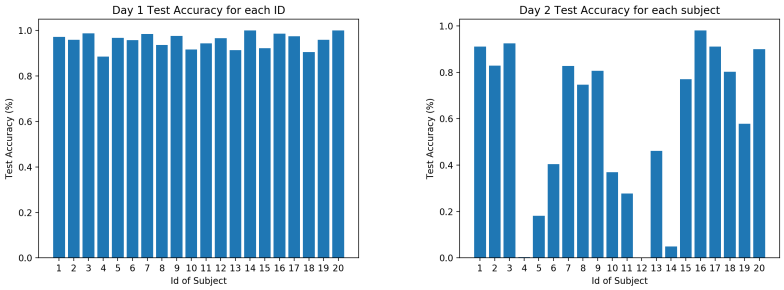
ICA: We perform Independent Component Analysis on the acceleration signal to find the independent source signals [19]. We then order the sources according to their kurtosis and input them into a 1D convolutional neural network.

TDEBOOST: We report the published results of Frank *et al.* who collected the McGill dataset [7].

Baseline: We use a 1d convolutional neural network on the three dimensional

acceleration signal. This is our baseline since it is the most straightforward approach.

Statistical features: We construct a feature vector using the mean, standard deviation, range, and energy of the x, y, z acceleration signals. We perform classification using a fully connected neural network.



(a) Day 1 Test Accuracy per Subject (b) Day 2 Test Accuracy per Subject

Figure 13: Test Accuracy for each subject

We note that the test errors were not uniformly distributed over all subjects, but certain subjects were significantly harder to classify on Day 2. In particular, subject 4 changed from a loosely flowing dress to tight shorts and subject 14 changed from jeans to baggy shorts [7]. In Figure 13, we show the test accuracy for each subject when using the magnitude signal as input to a 1D convolutional neural network.

From our results in Table 2, we see that rotation-invariant techniques are essential for high accuracy on the Day 2 data where sensor orientation is changed. Neither "Baseline" or "Statistical features" account for rotation-invariance and consequently perform worse than the rest of the methods for the Day 2 data. However, by assuming a fixed sensor orientation, they are able to leverage additional information about the direction of acceleration. This allowed them to

outperform all other methods on the Day 1 data where the sensor orientation remained constant.

The highest performing methods on Day 2 all used 2D feature representations and leveraged the more sophisticated classification method of 2D convolutional neural networks. However, this performance gain is at the cost of computational efficiency. Waveform-based feature representations such as "Magnitude", "PCA", and "ICA" have a much smaller computational footprint in both the feature extraction and classification stages, but performed worse than more complex methods using feature images.

While the continuous wavelet transform proved to be most effective, our novel method of fractional order derivative images (FODI) outperformed all previously published research results on classification of the McGill Dataset. This is an encouraging result and suggests that FODI's can be used as an alternative method for extracting 2D feature representations from 1D waveforms.

7 Conclusion

In this thesis, we have thoroughly discussed each stage of the machine learning pipeline for classifying time-series waveforms. In particular, we have focused on applications to gait recognition with inconsistent sensor data and outlined many approaches to mitigate the effects of sensor-induced factors. We first examined effective preprocessing techniques to resample, denoise, and align our acceleration signals. We then enumerated various ways to transform our three-dimensional signal into components that are either geometrically or statistically invariant to sensor rotation. Next, we described methods to increase feature complexity by transforming our waveform into a rich 2d feature representation. Finally, we introduced state-of-the-art classification techniques using convolu-

tional neural networks and evaluated the performance of each feature-extraction method on the McGill dataset.

We discussed both well-known methods and recent experimental approaches, and also proposed our own novel technique of using fractional order derivatives for feature extraction. To our knowledge, this is the first use of fractional order derivatives in this context. We found that our proposed method surpassed previously published results and achieved comparable test accuracy to industry-standard techniques such as the continuous wavelet transform.

There is still a long way to go for accelerometer-based gait recognition to be adopted as a universal form of biometric security. To compete with current forms of biometric security we expect gait recognition to reach an accuracy rate above 95%. For comparison, Facebook's DeepFace achieved a 97.5% accuracy for facial recognition on the Labeled Faces in the Wild (LFW) dataset [30] and fingerprint is accurate up to 99% according to a study by NIST [32]. However, accelerometer-based gait recognition is not far behind more resource-intensive forms of gait recognition such as camera-based methods. Rida *et al.* achieved the state-of-the-art accuracy of 86.5% for camera-based gait recognition on the CASIO Gait Dataset B [31]. Of course there are many different factors that make it unfair to directly compare these results. We only include these numbers to have a general perspective regarding the state of the field.

In the future we would like to scale our methods to a greater number of subjects. In order for gait recognition to be employed in real-world use cases, we must be able to identify between a far greater number of subjects than the twenty subjects of the McGill dataset. Since convolutional neural networks can be easily scaled to handle more complex learning problems, we believe our proposed pipeline will maintain high performance for a larger number of subjects. We hope to test this using larger datasets such as the Osaka dataset, which

contains walking data for 744 different subjects [29].

Although our results are specific to gait recognition, these general techniques can be utilized in any machine learning application with periodic time-series data. In addition, the methods for rotation-invariant feature extraction can apply to any sensor data that has spatial dimensionality. We are interested to pursue in the future how these techniques will translate for applications beyond gait recognition.

Acknowledgements

We would like to thank Marina Neseem for helpful discussions regarding deep learning as well as BahaaAlDeen AboAlNaga for his contributions to the implementation of fractional order derivatives. We would also like to thank Professor Adel Belouchrani for sharing his expertise in independent component analysis.

References

[1] Zhong, Yu & Deng, Yunbin. (2014). Sensor Orientation Invariant Mobile Gait Biometrics. IJCB 2014 - 2014 IEEE/IAPR International Joint Conference on Biometrics. 10.1109/BTAS.2014.6996246.

[2] Kobayashi, Takumi & Hasida, Koiti & Otsu, Nobuyuki. (2011). Rotation invariant feature extraction from 3-D acceleration signals. 3684-3687. 10.1109/ICASSP.2011.5947150.

[3] Gadaleta, Matteo & Rossi, Michele. (2016). IDNet: Smartphone-based Gait Recognition with Convolutional Neural Networks. Pattern Recognition. 74. 10.1016/j.patcog.2017.09.005.

[4] Subramanian *et al.* (2015). Orientation invariant gait matching algorithm based on the Kabsch alignment. IEEE International Conference on Identity, Security and Behavior Analysis, Hong Kong, 2015, pp. 1-8.

[5] Zhao, Yongjia & Zhou, Suiping. (2017). Wearable Device-Based Gait Recognition Using Angle Embedded Gait Dynamic Images and a Convolutional Neural Network. Sensors. 17. 478. 10.3390/s17030478.

[6] Sprager, S. & Juric, M.B. (2015). An Efficient HOS-Based Gait Authentication of Accelerometer Data. IEEE Trans. Inform. Forensics Secur. 2015, 10, 1486–1498.

[7] Frank, J. & Mannor, S. & Pineau, J. & Precup, D. (2013). Time series analysis using geometric template matching. IEEE Trans. Pattern Anal. Mach. Intell. 2013, 35, 740–754.

[8] Vahdatpour, A., & Amini, N., & Xu, W., & Sarrafzadeh, M. (2011). Accelerometer-based on-body sensor localization for health and medical monitoring applications. Pervasive and mobile computing, 7(6), 746–760. https://doi.org/10.1016/j.pmcj.2011.09.002

[9] Bayat, Akram, & Pomplun, Mark, & Tran, Duc. (2014). A Study on Human Activity Recognition Using Accelerometer Data from Smartphones, Procedia Computer Science, Volume 34, 2014, Pages 450-457, ISSN 1877-0509.

[10] Middleton, L. & Buss, A. &, Bazin, A. & & Nixon, M.S. (2015). A floor sensor system for gait recognition. Fourth IEEE Workshop on Automatic Identification Advanced Technologies (AutoID'05), pp. 171-176.

[11] Hu, M. & Wang, Y. &, Zhang, Z. & Zhang, D., & Little, J.J. (2013). Incremental Learning for Video-Based Gait Recognition With LBP Flow. IEEE Transactions on Cybernetics, vol. 43, no. 1, pp. 77-89.

[12] Wiener, Norbert. (1930). Generalized Harmonic Analysis. Acta Mathematica. 55: 117–258.

[13] Emery, James. (2011). The Cubic Spline. http://www.stem2.org/je/cs.pdf.

[14] Wan, Changsheng & Wang, Li, & Vir, Phoha. (2018). A Survey on Gait Recognition. ACM Comput. Surv. 51, 5,Article 89 (August 2018), 35 pages.

[15] Hyvärinen, A. & Oja, E. (2000). "Independent component analysis: Algorithms and applications" (PDF). Neural Networks. 13 (4–5): 411–430. CiteSeerX 10.1.1.79.7003. doi:10.1016/S0893-6080(00)00026-5. PMID 10946390.

[16] Folland, Gerald B. (2002). Advanced Calculus, p. 193, Prentice Hall.

[17] Alexis, Kostas. www.kostasalexis.com/inertial-sensors.html.

[18] Barralon, P. & Vuillerme, N. & Noury, N. (2006). Walk detection with a kinematic sensor: Frequency and wavelet comparison. Proceedings of the 2006 International Conference of the IEEE Engineering in Medicine and Biology Society, pp. 1711–1714.

[19] Mantyjarvi, J. & Himberg, J. & Seppanen, T. (2001). Recognizing human motion with multiple acceleration sensors. 2001 IEEE International Conference on Systems, Man and Cybernetics. e-Systems and e-Man for Cybernetics in Cyberspace (Cat.No.01CH37236), pp. 747-752 vol.2.

[20] Hyvärinen A. (2012). Independent component analysis: recent advances. Philosophical transactions. Series A, Mathematical, physical, and engineering sciences, 371(1984), 20110534. https://doi.org/10.1098/rsta.2011.0534

[21] Butterworth S. (1930). On the Theory of Filter Amplifiers. Wireless Engineer vol. 7, pp. 536–541.

[22] Glowinski S., & Blazejewski A., & Krzyzynski T. (2017). Human Gait Feature Detection Using Inertial Sensors Wavelets. Wearable Robot. Chall. Trends. DOI: 10.1007/978-3-319-46532-6_65.

[23] Martin, E. (2011). Novel method for stride length estimation with body area network accelerometers. IEEE Topical Conference on Biomedical Wireless Technologies, Networks, and Sensing Systems, pp. 79-82.

[24] Ji, N., & Zhou, H., & Guo, K., & Samuel, O. W., & Huang, Z., & Xu, L., & Li, G. (2019). Appropriate Mother Wavelets for Continuous Gait Event Detection Based on Time-Frequency Analysis for Hemiplegic and Healthy Individuals. Sensors (Basel, Switzerland), 19(16), 3462. https://doi.org/10.3390/s19163462.

[25] Han, D. & Renaudin, V. & M. Ortiz. (2014). Smartphone based gait analysis using STFT and wavelet transform for indoor navigation. International Conference on Indoor Positioning and Indoor Navigation (IPIN), Busan, pp. 157-166.

[26] Renaudin, V. & Susi, M. & Lachapelle, G.. (2012). Step length estimation using handheld inertial sensors. Sensors (Basel, Switzerland). DOI: 10.3390/s120708507.

[27] Nyan, M.N. & Tay, Francis & Seah, K.H.W. & Sitoh, Y.Y.. (2006). Classification of gait patterns in the time–frequency domain. Journal of biomechanics. 39. 2647-56. 10.1016/j.jbiomech.2005.08.014.

[28] Sejdic, Ervin & Djurovic, Igor & Jiang, Jin. (2009). Time–frequency feature representation using energy concentration: An overview of recent advances. Digital Signal Processing. 19. 153-183. 10.1016/j.dsp.2007.12.004.

[29] Ngo, T.T. & Makihara, Y. & Nagahara, H. & Mukaigawa, Y. & Yagi, Y.. (2014). The Largest Inertial Sensor-based Gait Database and Performance Evaluation of Gait Recognition. Pattern Recognition, 47(1), pp. 228–237.

[30] Taigman, Y. & Yang, M. & Ranzato, M. & Wolf, L.. (2014). DeepFace: Closing the Gap to Human-Level Performance in Face Verification. 2014 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1701-1708.

[31] Rida, I. & Jiang, X. & Marcialis, G. L.. (2016). Human body part selection by group lasso of motion for model-free gait recognition. IEEE Signal Processing Letters, vol. 23, no. 1, pp. 154–158.

[32] "NIST Study Shows Computerized Fingerprint Matching Is Highly Accurate." NIST, 27 Nov. 2017, www.nist.gov/news-events/news/2004/07/nist-study-shows-computerized-fingerprint-matching-highly-accurate.