Fall Detection Using Neural Networks

Warren Zajac
Sheridan College, zajacw@sheridancollege.ca

FALL DETECTION USING NEURAL NETWORKS

A Thesis

Presented to

The Faculty of Applied Science and Technology, School of Applied Computing

of

Sheridan College, Institute of Technology and Advanced Learning

by

Zajac, Warren

In partial fulfillment of requirements

for the degree of

Honours Bachelor of Computer Science (Mobile Computing)

December, 2020

ABSTRACT

FALL DETECTION USING NEURAL NETWORKS

Warren Zajac
Sheridan College, 2020

Advisor:
Professor Dr. Khaled Mahmud

Falls inside of the home is a major concern facing the aging population. Monitoring the home environment to detect a fall can prevent profound consequences due to delayed emergency response. One option to monitor a home environment is to use a camera-based fall detection system. Conceptual designs vary from 3D positional monitoring (multi-camera monitoring) to body position and limb speed classification. Research shows varying degree of success with such concepts when designed with multi-camera setup. However, camera-based systems are inherently intrusive and costly to implement. In this research, we use a sound-based system to detect fall events. Acoustic sensors are used to monitor various sound events and feed a trained machine learning model that makes predictions of a fall events. Audio samples from the sensors are converted to frequency domain images using Mel-Frequency Cepstral Coefficients method. These images are used by a trained convolution neural network to predict a fall. A publicly available dataset of household sounds is used to train the model. Varying the model's complexity, we found an optimal architecture that achieves high performance while being computationally less extensive compared to the other models with similar performance. We deployed this model in a NVIDIA Jetson Nano Developer Kit.

Keywords: Fall Detection, Machine Learning, Neural Network, SoC, Edge Processing, IoT

ACKNOWLEDGEMENTS

Throughout the writing of this thesis, I have received a great deal of support and assistance.

I would first like to thank my supervisor, Dr. Khaled Mahmud, whose expertise was invaluable in formulating the research questions and methodology. Your insightful feedback pushed this to a higher level.

I would like to acknowledge my colleagues from my internship at Tech4Life for their collaboration. I would particularly like to single out my supervisor at Sheridan College, Tarek El Salti. Tarek, I want to thank you for your patient support and for all the opportunities to further my research.

In addition, I would also to thank my parents for their support and sympathetic ear. You are always there for me. Finally, I could not have completed this thesis without the help of my friends, Harrison U-Ebili and Cory DaSilva, who provided stimulating discussions as well as happy distractions outside of my research.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER ONE

INTRODUCTION

1.1: The Problem Context

Video recording of physical environments has become the standard for monitoring public areas. However, video monitoring technology is not being accepted into homes with the same level of trust in public locations. The information captured by a camera can detect an individual falling and can be a useful tool in assisting individuals living independently. According to Statistics Canada, the number of seniors in Canada is rising from the existing 12.6 million in 2001 to an expected 18.5 million by 2021 [1]. Falls are the most common cause of injury among older Canadians [2]. In 2008-2009, 20% of Canadians aged 65 or older reported a fall in the previous year (862,000 seniors) [2]. Many seniors (i.e., over 65) live alone, which dramatically increases their risk of being stranded after suffering from an undetected fall. Overall, 20 percent of men and 36 percent of women over 65 currently live alone. Seniors have a much higher risk of fall hazards and may suffer from additional medical/physical complications because of an undetected fall. In response to these issues, using in-home monitoring devices can help reduce hospital admissions. It can also increase the overall quality of life for seniors when a fall is detected and receive prompt medical attention [3].

While video-based solutions have met some success, this paper proposes using a new audio sensor approach using machine learning. Audio samples can bring contextual awareness into the surrounding environment by the data gathered — IoT based sensors offer edge processing and present instant decisions with real-time reaction to events. Preventive welfare checks are costly, and the research into monitoring can be used to improve the independence quality of life for our aging population. This can be done while reducing health care risks for those living at home, alone. This research proposes a machine learning algorithm that uses low-frequency audio sensors and a continuous data stream to evaluate real-time events.

The proposed algorithm will detect a fall of different objects and classify them based on the sensed input type and frequency. Additional sensors may be added to the system to improve the overall quality of detection and alerting system. Processing signals in real-time, without the issues surrounding cloud-based latency will also offer enhanced security and privacy. This provides the advantage of instantaneous decisions without the need for traditional cloud-based machine learning (ML) access. Using this approach to fall detection, the system would allow individuals to live in rural areas without the need for high speed, low latency internet access.

Figure 1 shows an example deployment of a multiple sensor data collection station with its home machine learning TPU and an optional internet uplink for cloud updates and global model verification.

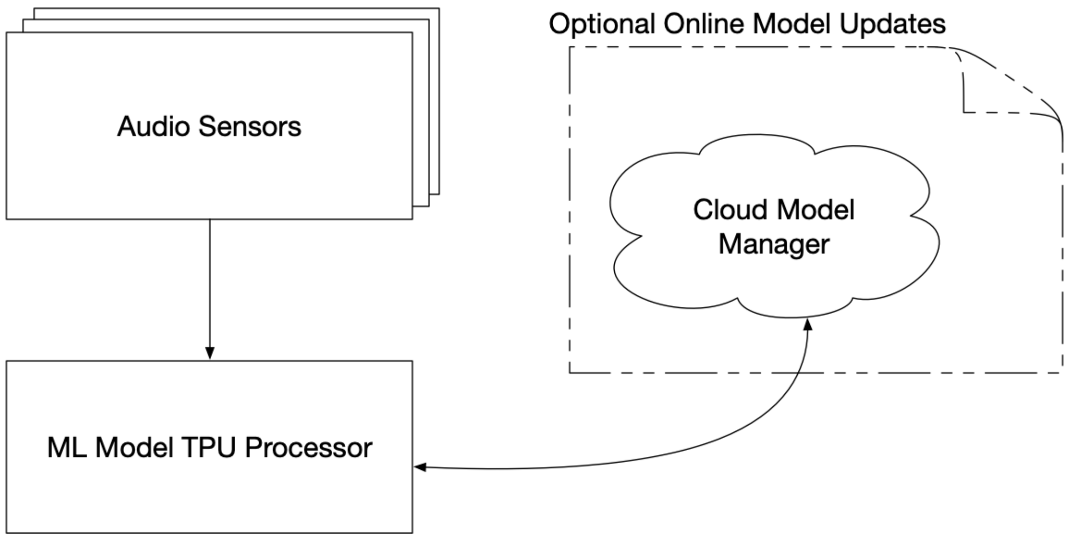


Figure 1: Multiple Sensor w/Optional Cloud Deployment

1.2: Definition of Terms

Term	Definition
MFCC	Mel-frequency cepstral coefficients
Wavelet	A wavelet is a mathematical function used to divide a function or continuous-time signal into different scale components.
Fourier Transform	The Fourier transform is an extension of the Fourier series that results when the period of the represented function is lengthened and allowed to approach infinity. Due to the properties of sine and cosine, it is possible to recover the amplitude of each wave in a Fourier series using an integral.
TensorFlow	TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library and is also used for machine learning applications such as neural networks.
Convolutional Neural Network (CNN)	A convolutional neural network is a class of deep neural networks, most commonly applied to analyzing visual imagery. They are also known as shift invariant or space invariant artificial neural networks, based on their shared-weights architecture and translation invariance characteristics.

PULP-NN	Parallel Ultra-Low-Power RISC-V Processors Accelerating Quantized Neural Networks
Bayesian Network	Bayesian networks are a type of probabilistic graphical model that uses Bayesian inference for probability computations. Bayesian networks aim to model conditional dependence, and therefore causation, by representing conditional dependence by edges in a directed graph
Support Vector Machine	Machine learning support-vector machines are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis.
Principal Component Analysis	The principal component analysis is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components.
Condensation Algorithm	The condensation algorithm (Conditional Density Propagation) is a computer vision algorithm. The principal application is to detect and track the contour of objects moving in a cluttered environment. Condensation is a probabilistic algorithm that attempts to solve this problem.
Particle Filtering	particle filtering method refers to the process of obtaining the state minimum variance distribution by finding a set of random samples propagating in the state space to approximate

	the probability density function and replacing the integral operation with the sample mean.
Fuzzy Logic	Fuzzy logic is an approach to computing based on "degrees of truth" rather than the usual "true or false" (1 or 0) Boolean logic on which the modern computer is based.
VGGNet	VGGNet describes a CNN model where the principal features of this architecture are several successive convolutional layers.

1.3: Statement of the problem

Using computer vision and standard security cameras to monitor sensitive areas of our environment is not ethical and has significant privacy concerns. Many locations that need monitoring are also susceptible to personal privacy. The elderly population is at the highest risk for complications due to falls, and many individuals are not living with someone who can help.

The National Electronic Injury Surveillance System was evaluated for emergency department injuries in patients over 60 years old. Observed in this system, from the estimated 3.4 million emergency visits, 7.2% of these were bathroom-related accidents [4]. Privacy needs to play a critical role in our safety within our own homes. The need for monitoring should not infringe on an individual's privacy. The current costs involved in creating a private and safe monitored environment are cost prohibited. Fall detection systems require custom installation of equipment into existing structures.

A monitoring solution is needed to permit the elderly to live in their existing homes and be safely monitored. With the mass production of security cameras and in-home surveillance, the equipment has become inexpensive and readily available. Many proposed systems offer a computer vision, video-based fall detection system. The main drawback of camera surveillance is the loss of privacy while being always watched by a camera. Video-based detection is optimal for its accuracy and low cost. However, it is very intrusive to the individuals who are in the home [1]. Video-based systems can suffer from a field of view issue where cameras are limited by the visible area within their sightline. If an object comes in between the camera and a fall event, the camera can miss it. Video monitoring of sensitive locations such as bedrooms and bathrooms also pose major privacy issues.

Wearable sensor technology has great protentional for detecting falls as well. Seniors using this technology will not have the same privacy concerns that come with a camera system. Seniors will need to manage and maintain the fall detection device actively.   Body located sensors provide high accurate predictions. Bourke [5] and Chen [6] both proposed a fall detection method by deploying wearable accelerometer sensors to be worn by seniors. Wearable sensors provide very high sensitivity for fall detection. Many issues come into play with operating these devices, as users may forget to wear their sensors or forget to recharge the devices. The Apple Watch Series 4 is one of the first Apple wearables to provide fall detection. However, the watch must be removed and recharged every day (~18 hours) [7] to be effective. The main drawback of these devices is typically the overall battery and charging design.

1.4: Purpose of the Study

Find alternative means of detecting environmental events using small, inexpensive sensors, sensing events without the need for computer vision or a security-based camera system.

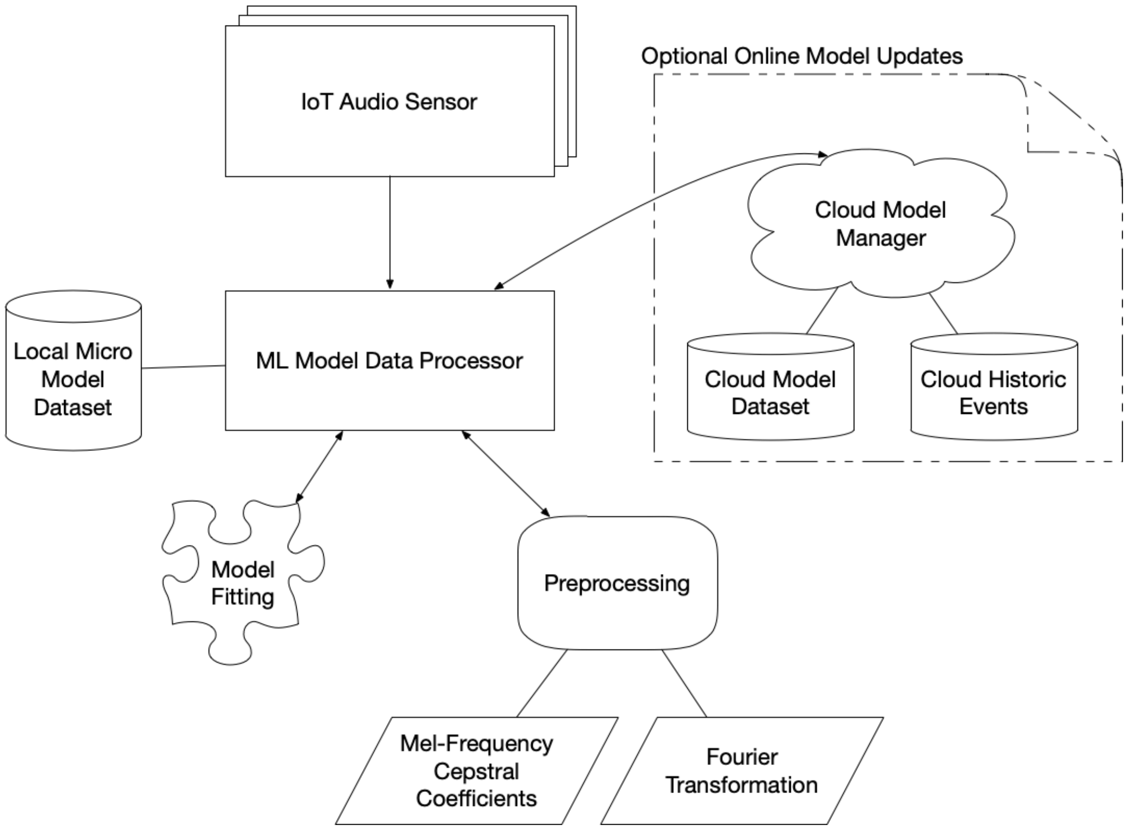


Figure 2: Fall Detection ML life cycle

In this research, we explore the possibility of using audio features to monitor the local environment and predict what events are occurring. This type of monitoring has the advantage of not requiring a direct field of view, like camera systems. Using multiple sensors, the system can make accurate environmental predictions with only audio information. Audio sensors monitor room acoustics using edge processing ML models, allowing immediate predictions to be made. The system will function without the need for complex hardware or highspeed internet access. Easier deployment of inexpensive sensors replaces existing more expensive provided provide overall privacy level privacy for the individuals. Using a learning system model life cycle (as shown in Figure 2), the system collects data, processes it locally, and then uploads new details to the cloud or local infrastructure. This model information is used to update the global model, allowing for an increased overall accuracy of all connected systems. The iterative learning of the model ensures the system is always improving itself over time.

1.5: Thesis Statement

Instead of using expensive and intrusive camera-based fall detection system, we can use acoustic sensor-based fall monitoring system where sound samples are converted to frequency images. We can train a convolutional neural network to predict the event of a fall from such image input and achieve comparable performance. With proper selection of layers in a CNN architecture, we can optimize the desired performance parameters of the model suitable for fall detection application.

1.6: Motivation

Utilizing multiple inexpensive sensors to collect data and provide an accurate context-aware ML model running on a low-cost local system. This approach protects personal data from streaming into the cloud and permits our aging population to live without the worry of complex welfare check-in systems. Current wearable technology is based on HARD falls, such as the Apple Watch. Wearables offer many new features to monitoring; however, humans can easily defeat wearables by not wearing them, either intentional or by forgetting to put the device back on. Using an alerting system based on sensors that do not require HCI provides a robust alternative that does not require modification of the senior's existing habits. In Canada, many seniors (over 65) live alone (20 percent of men and 36 percent of women), where there is no one to perform welfare checks in an emergency. Seniors have a much higher risk of fall hazards; a monitoring system can also be used expanded to possibly allow the detection of symptoms that may arise without the user noticing them (habitual changes). Many existing technologies have been studied to monitor and alert events, but these are mainly devices that are push/panic systems and offer nothing if the user is incapacitated. Solutions that do work without interaction are some existing camera solutions, but these feel intrusive to individuals being monitored.

1.7: Contribution of the Study

Simple in-house deployment of the primary ML model using samples from low-frequency pressure and sound sampling around parts of the home permits the system to detect falls and accidents without the need for camera-based systems. The use of ambient sensors has also provided new methods for fall detection, and existing floors can utilize new designs using floor vibration sensors. Floor sensors offer a way for passive fall detection. While underfloor systems do not have the same drawbacks as wearable sensors, these sensors tend to be cost-prohibitive, and installations are often invasive. Modifications to the building would be costly and invasive. The locations monitored are also only available for each floor where these sensors have been installed.

CHAPTER TWO

A LITERATURE REVIEW

2.1: Introduction

Fall detection has previously been researched in many different ways. Devices are used to gather data and determine if a fall is categorized into ambulatory devices and passive devices. The use of invasive devices consists of devices that must be maintained by the end-user. Passive systems offer the ability to operate with minimal maintenance and overall user interaction. Ambulatory devices are typically created with the use of accelerometers, gyroscopes, goniometers, pedometers, and accelerometers.

2.2: Sensors for environmental monitoring

Static passive devices for monitoring falls consist of video camera floor sensors, IR sensors. The data is collected and then classified as an ML model to determine if an event has occurred. Many methods have been used to create classifications for falling prediction, supervised learning, and neural networks. This result will help the processing of the system's overall design and detection of the system. Prior research using accelerometers to monitor floor vibration and a microphone to listen for an abrupt loud event allowed the researchers to use energy calculation on the noises and vibrations that resulted in a normal statistical distribution for ambient events. They then used a Bayesian classification algorithm to determine if a fall event was detected versus other events. Using another Bayesian type classifier, the relevance of a vector machine was utilized by Jiang [17] to classify postures obtained from a video after a fall is to have thought to have occurred. They were using this as a reinforced position of verifying that an event occurred. They need to use additional sensors. The researchers did not use audio to image translation to provide more detailed analysis for an event based on just the audio signal as will be performed by our ML model.

2.3: Dataset Selection

The audio samples used in this process were from the dataset A3fall-v2.0 (used in previous works by D. Droghini et al. [8]). This dataset was selected for its sample size and variations on environment recordings for fall events. The dataset includes samples with and without noise added to them. Noisy samples provide a validation method and help us achieve a real-world type of analysis for fall detection events.

2.4: Model Selection

While a variety of research using support vector machines with training models have been widely used. A key reason for machine learning fitting and overall accuracy of the device within the ML Platform. In many cases, a heuristic decision tree was also used to assist the decisions of the model. Importantly, SVM is by far the least computationally intensive technique, compared to a single minimax probability machine. The SVM does not have the same accuracy when it comes to visual image detection and classification. While it has the least computational overhead, the DNN (deep neural network) will be the machine learning model that we use in our analysis. Fuzzy logic was another approach used to find outliers with the use of video fall detection. The addition of multiple cameras in video detection has been proposed to create 3D renderings of a person to produce more accurate results. Increased camera numbers increase complexity and a combination of video to create a live 3D model is computationally expensive. To increase the algorithmic accuracy, focusing on head tracking motion history was essential for fall detection. This method combined with a particle filter based on the algorithmic condensation system combined with the head's velocities, and that once passed a critical point detect a fall event. This approach requires several cameras and a heavy processing engine to process the data in real-time. PULP-NN demonstrated that coupling optimized software libraries with a parallel ultra-low-power computing platform achieves energy proportionality. Unlike commercial ARM-based solutions, it did not have to trade performance with energy efficiency, paving the way to fully software programmable Convolution Neural Networks (CNN) inference at the edge of IoT. [9] The overall success of Deep Learning (DL) has paved the way for many different DL deployments on embedded computing platforms for all kinds. Recent insights into extreme-edge and IoT end-nodes for the demand of IoT edge nodes. Cloud architecture [10] Wireless sensor networks (WSNs) consist of many smart sensors with limited computing, storage, communication, and energy resources. [11]

VGGNet Convolutional Neural Network is a model developed in 2014 by the VGG (Visual Geometry Group) group of Oxford University. The VGGNet model won second place in the 2014 ILSVRC. This model has excellent feature extraction and based on its performance, outperformed GoogleLeNet at the time [12].   Each VGGNet structure has a different version, depending on the total number of layers. VGGNet uses the increase in hidden layers to improve the prediction accuracy of its models [13].

2.5: Processing Sensors

Health monitoring systems can be set up to assure the safe operation of structures and require linking sensors with computational tools to interpret sensor data in structural performance. Although intensive development continues on innovative sensor systems, there is still considerable uncertainty in deciding on the number of sensors required and their location placement to obtain adequate structural behavior information. The characterization of a real structure's dynamic behavior is possible only if a minimum amount of information is available. This, in turn, implies that a minimum number of sensors must be placed on the structure under assessment. [14] Typical environmental sensing requires that sensors be placed in optimal sensor locations on a real structure. [14] Using a transformation function to study low-frequency vibrations could detect the effects of frequency on buildings. In Equation 1, parameters Vt, A, and B are used in this vibration function to compute the regression analyses of the measured vibrations from the data acquired by the sensors. [15]

Equation 1

$$\mathrm{COV}(F_v) = \frac{\sigma_{F_V}}{\overline{F}_V} = \sqrt{\sum_{i=1}^{n} \left(\frac{(Y_i - F_V(S_i, D_i))}{F_V(S_i, D_i)} \right)^2 \Big/ (n-1)},$$

Using this model to study the effect of low-frequency vibrations traveling through materials in a building, the study measured the impact of vibrations from railway cars on nearby neighborhoods. [15]. Studying the behavior of low-frequency vibrations during a fall event detection will also provide the ability to calculate the intensity of the impact overall using this regression model. Ball et al. used acoustics detection to detect cracks in machinery before any vibration signals could see the fault condition. Vibration analysis is already used in current machinery diagnostics. The additional use of acoustic signals to affective detect earlier fault detection was proposed. Using a method of acoustic signal capture, the system could detect earlier when there was an issue with the equipment's operation. The auditory signals produced were unique and could be easily classified based on the defect model. [16] This method of using sound waves to detect movement and pressure changes allows the detection of events that occur sooner and without the need for many sensors deployed over every inch of a surface.

2.6: Preprocessing Algorithm

Audio data pre-processing during sampling using the Fourier transform allows the system to move data from a time to a frequency domain with sines and cosines as the basis functions. This provides the average characteristics of a signal. The wavelet transform is used to move data from a space to a scale domain with the wavelets as the primary function giving the original signal's localized features. Wavelets occur in sets of families of functions, and each is defined by dilation, which controls the scaling parameter, and translation, which controls the position of the kernel in time. In continuous wavelet transform (CWT), time t and time-scale parameters vary continuously. The CWT of a signal x(t) is defined as:

Equation 2

$$CWT_x(a,b) = \sqrt{a}\int_{-\infty}^{\infty} x(f)H^*(af)\,\mathrm{e}^{(j2\pi fb)}\,\mathrm{d}f\,.$$

Where x(f) and H(f) are the continuous-time Fourier transforms of x(t) and h(t), respectively. The calculations are based upon octave band analysis in which each octave is equally subdivided into voices. The calculation of the transform is achieved by implementing the discrete version of the equation. [17] Unsupervised Disaggregation of Low-Frequency Power Measurements was based on research efforts that have attempted to improve "Non-Intrusive Appliance Load Monitoring" (NALM). Often this has been through proposing alternative signature identification techniques. Farinaccio and Zmeureanu [18] use a pattern recognition approach to disaggregate whole-house electricity consumption into its major end-uses. Using this approach, the slight variations in audio signal recordings can offer a unique signature to detect discrete events without multiple sensors. [19] et al. proposes a neural net approach for identifying the electrical signatures of residential appliances. Rabiner et al. suggest collecting data at higher frequencies (e.g., 8,000 Hz) to use higher harmonics to generate appliance signatures. Using various pre-processing techniques, the system could be tuned into the classification of multiple points of interest. Here we will be focused on the lower frequencies generated during a fall event. Still, the model only needs minor changes to the pre-processing to be used to classify other events. Using an integer programming approach, [20] used this approach to disaggregate residential power use. This permitted the extracting of a total of nine features from the measured current signal and successfully used them to classify the operational state of an appliance. While [21] describes an electric appliance recognition method. It uses Principal Component Analysis (PCA) to extract features from electric signals. These features are classified using a Support Vector Machine. For unregistered appliances, a one-class SVM was used. [22] use a dynamic Bayesian network to consider user behavior and a Bayesian layer to disaggregate the data online. However, these methods have practical limitations that motivate the development of alternative techniques. Matthews et al. reflect on some of these works and describe a workable solution [23, 21]. In comparison, the overall work focuses on the disaggregation of low-frequency signals for fall detection without the need for intrusive sensors for things like computer vision. Several research efforts have prototyped methods for disaggregation of signals, proving its overall viability [24] Stable-state signatures relate to more sustained changes in power characteristics when an appliance is turned on/off. These persist until the state of the appliance changes, which can be captured with low-frequency sampling. The low-frequency sampling technique can be further used to sample audio events and environment state changes. Even for stable-state features, the sampling frequency is important since, at low sampling rates, the probability of multiple independent events will occur. Using this approach, these low-frequency sampling rates can be used for the identification of signals with disaggregating and identification of the events in real-time. In pervasive computing, intelligent environments that identify and track occupants' behavior in a room are gaining immense attention. The methods can be applied in homes for the elderly and disabled as an enabling technology for monitoring and surveillance systems. While hidden video cameras are being used to identify people and their motion, the feeling of big brother watching is not that pleasant for patients in a medical facility or shoppers in a departmental store. [25] While our proposed audio classification system does not have the overall negative stigmatism of video monitoring and can be more effective in detecting events even off-camera (out of view).

CHAPTER THREE

METHODOLOGY

3.1: Introduction

In the subsequent sections, we will be covering the individual components of the system. The system has been split into two sections, the training, and the deployment. The design centers around the chosen CNN model of the overall system. The system uses audio sounds to predict events based on small sample recordings. The training of the models was performed using the A3Fall audio dataset. This dataset includes individual sound samples passed through a filtering process and converted into unique MFCC images. Training and testing for the various models used these images.

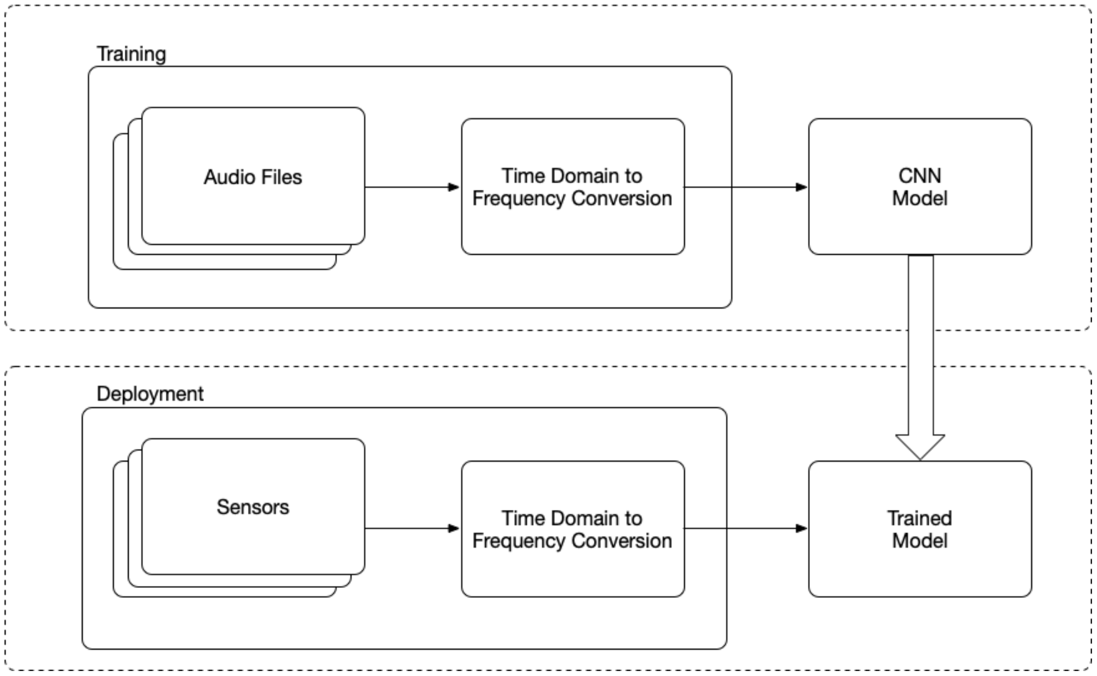

Training
Audio Files
Time Domain to Frequency Conversion
CNN Model
Deployment
Sensors
Time Domain to Frequency Conversion
Trained Model

Figure 3: System data flow

Figure 3: System data flow, we show how the overall audio data flows through the entire system. Showing the process of training the initial model and finally deploying it.

Using the audio dataset, audio recordings were converted into MFCC images and stored in their own subdirectories. The samples were split into an 80/20 (train/test) split. In the following sections, we will discuss the individual elements in detail. This solution will not require a cloud-based system architecture for processing the classification probabilities. The entire solution can be operated off-grid and even battery-powered for demonstrational purposes.

3.2: Audio Samples

The model was trained using a variety of audio samples from the A3fall dataset. The audio recordings in this dataset include multiple microphones and even samples from a unique FAS sensor.   The FAS sensor's unique design is believed to provide an increase in accurate fall sound recordings than the regular microphones. This sensor uses an enclosure with a chamber for the microphone. Recording audio in the FAS enclosure will amplify low frequencies while physically canceling out higher ones. This enclosure offers a physical filter for the fall detection system and does not require any special pre-process filtering of that signal.

The dataset comprises recordings of fall events related to everyday objects and "Rescue Randy," a mimicking human doll by Simulaids. The items and the number of files per object are listed in Table 1.

Table 1: Composition of the A3Fall-v2.0 Dataset

Classification	Room A	Room B	Room C
Basket	64	40	40
Fork	64	40	40
Ball	64	40	40
Book	64	40	40
Bag	64	40	40
Chair	96	40	40
Table	0	40	40
Guitar Slide	0	40	40
Nipper	0	40	40
Keys	0	40	40
Hook	0	40	40
Coat Hook	0	40	40
Manikin Doll	44	0	0
Human Fall	0	40	40
Daily Life Activities	2530	9055	5550

Objects were dropped at four distances from the sensors, 1 m, 2 m, 3 m, and 4 m, and with different angles. Except for the chair and the basket, which have been overturned from their natural position, half of the falls were performed at the height of 0.5 m and the other half at 1 m. The doll was dropped from an upright position and from in a chair [8].

The entire database was recorded using the Presonus AudioBox 44VSL soundcard, and the AKG C400 BL microphone was used both in the array that in the FAS. Two loudspeakers were used to reproduce the classical and rock audio tracks, then captured by the FAS and the array microphone, then digitally added (mixed) to individual fall events.

The Floor Acoustic Sensor (Olivetti, P. (2015). Italian Patent 0001416548, July 1) comprises a resonant enclosure and a microphone located inside it. At the bottom of the chamber, a membrane maintains direct contact with the floor to guarantee the surface's acoustic coupling. The inner container accommodates the microphone and is where the acoustic resonance phenomenon takes place. A layer of acoustic isolation material can cover it, and it is enclosed by the outer container that further reduces the intensity of the acoustic waves that propagate through air. The enclosure has been manufactured in Polylactic acid with a 3D printer; its diameter is 16.5 cm, and its height 5.5 cm [8].

The microphone, an AKG C 400 BL, was inserted in the enclosure. The microphone's outer case was removed to extract the capsule that was inserted in the sensor enclosure. The AKG C 400 BL is characterized by a hyper-cardioid directivity pattern. Thus, it has been oriented so that the maximum gain is towards the floor [8].

3.3: Algorithms for Processing

In this section, we will be discussing all the algorithms being used. Audio to image conversion and the transformation for the model's CNN image recognition component.

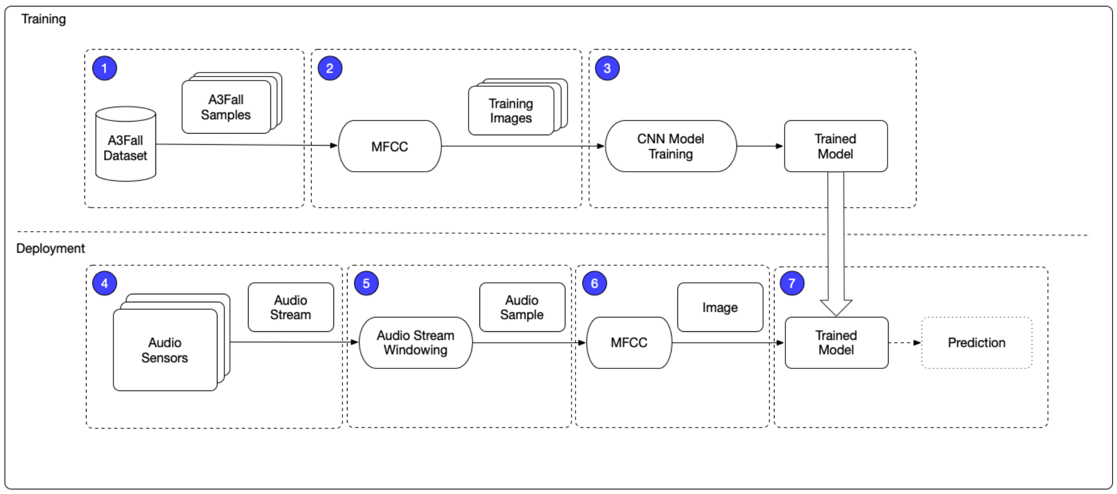


Figure 4: Detailed data flow

3.3.1: Audio Dataset

In section 1 of Figure 4, we use all the labeled samples from the A3Fall dataset to build a testing/training set. This dataset is split into 10,000 randomized audio samples with a training/testing 80/20 split. We will save the processed samples as images to reduce training time between the various research models. The final model will be used in the last proposed deployment, as seen in Figure 4.

3.3.2: Pre-processing of Audio in Frequency Domain

Figure 4, section 2, shows that the sound samples are converted into images using the Mel-Frequency Cepstral Coefficients (MFCC) pre-processing. Each sound from the dataset is decomposed into overlapping frames (windowing). The data from this process is then transformed using the fast Fourier transform function. The resulting data then gets filtered through the Mel filters (converting the data to the Mel scale) and converter to the cepstral coefficients. The entire process creates an image that can then run on the CNN (Convolutional Neural Network) for image identification and classification. The overall process is visualized in Figure 5.

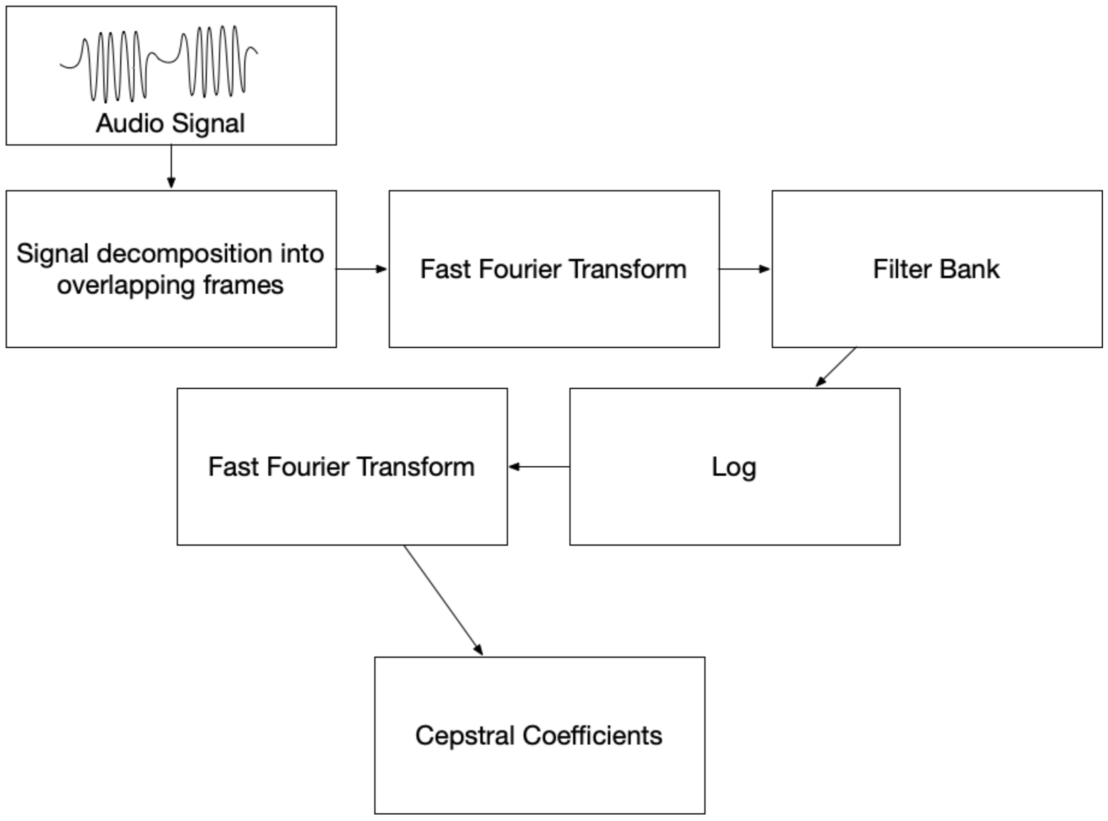


Figure 5: Mel-Frequency Cepstral Coefficients Processing

Mel-frequency spectrum is a nonlinear Mel-scaled representation of a sound's linear cosine transformation of a short-term power spectrum. Mel-frequency cepstral coefficients are widely used features in sound recognition [26]. The process to extract these MFCC features requires several steps to achieve this; they are outlined in Figure 5. First, the windowing of a given signal is performed and is divided into n overlapping samples. After that, the application of a Fast Fourier Transform to all of these samples creates their spectrogram. Then, the data is scaled using the Mel-frequency filter to obtain the Mel-frequency spectrum. Finally, the spectrum's log is taken, and using a Fast Fourier Transform gives the final Mel-frequency cepstral coefficients.

The following figure is an example of a recorded acoustic audio sample from the A3Fall dataset. The figure shows the sound energy from the event of a ball dropping. The audio frames in the dataset are between 0.5 and 1 second in length.

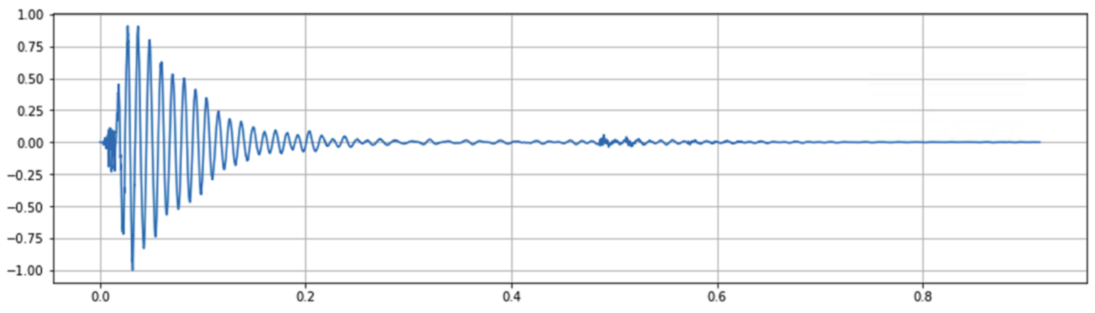


Figure 6: Raw Audio Signal

Once the audio has been loaded into memory, a series of filters are processed on the acoustic information before converting to an MFCC image. The first step in the process is the windowing method. Here the Kaiser windowing method was used in the filter process. Below is a plot of the Kaiser window. Different windowing options were used to filter the audio signal. The Kaiser windowing provided a well-balanced filter of low and high-frequency signals.

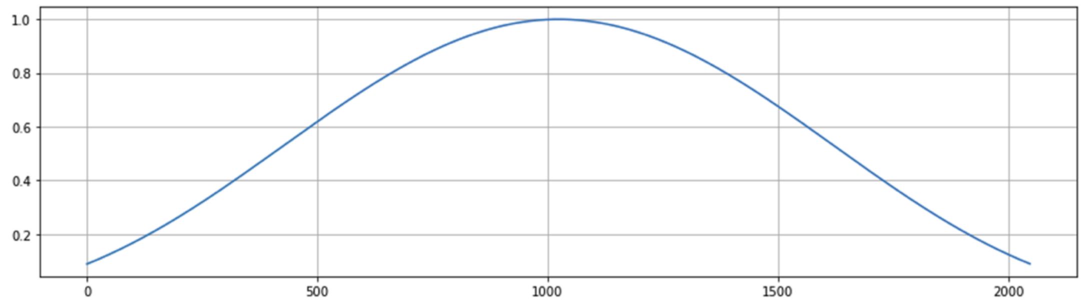


Figure 7: Kaiser Window

After the windowing has been created, the audio data frame is filtered through this windowing, creating a new data frame. In the following figure, Figure 8, the difference between the original audio frame and the new filtered audio frame after windowing is shown.

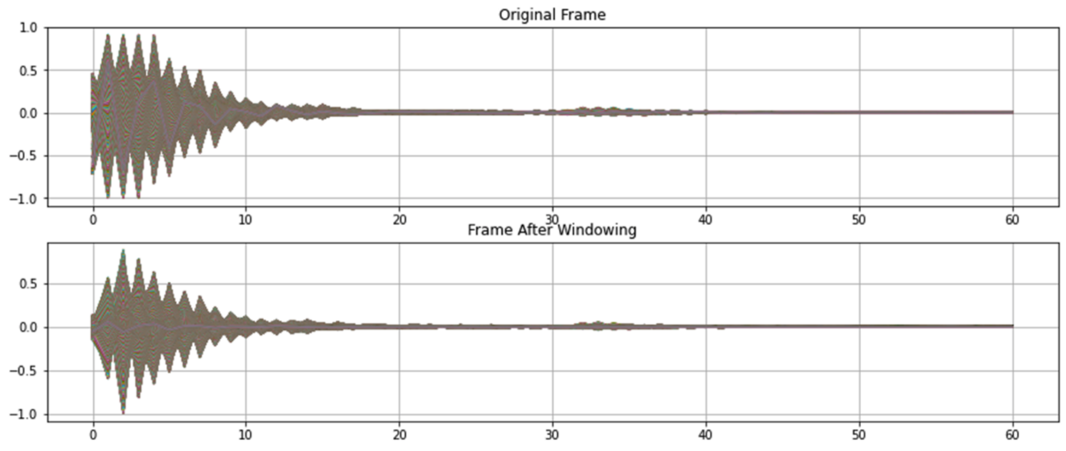


Figure 8: Pre/Post Windowing

Once windowing has been completed, the MFCC is generated. In the following figures, the MFCC can be visualized before and after the logarithmic scale is applied. The logarithmic scaling of MFCC represents how humans perceive audio sounds and help identify audio features for classification.

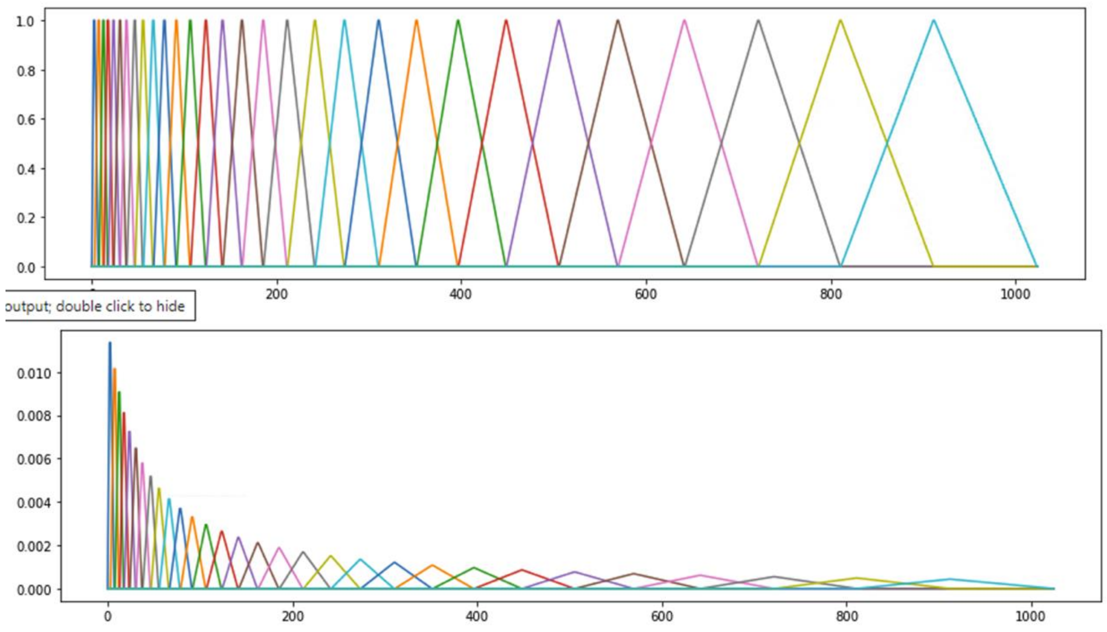


Figure 9: MFCC

Once the audio file has been processed through the filters and MFCC, the final MFCC image is shown below. Each image of the audio sample will be saved, and this image will be used for training and testing. Saving these images and reusing them during the various model tests is how the CNN model's analysis was performed.

The following spectrograph shows the audio in different bands to help distinguish the detail between them. During each audio capture and analysis, the data is extracted from the microphone, pre-processed, and then passed into Librosa's MFCC function, which generates an MFCC from the time series audio data.

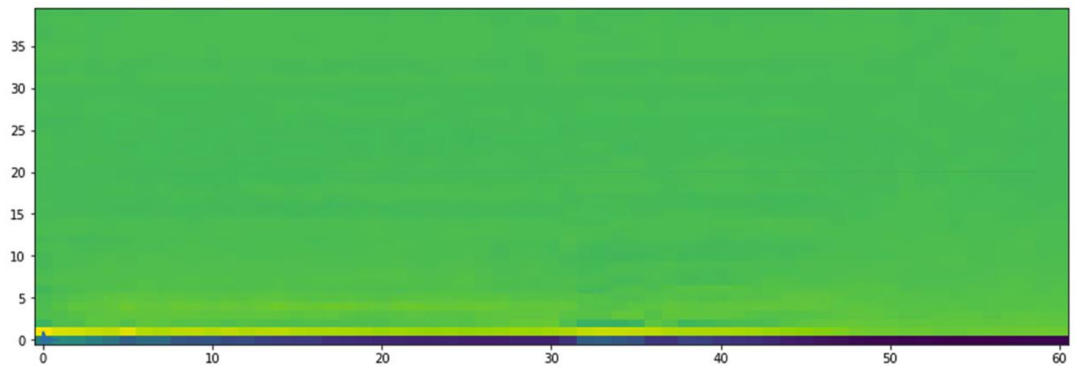


Figure 10: Final MFCC of Audio Sample

The MFCC Image in Figure 10 is a visual representation of the data the model is trained and tested. The model will categorize each sample into a detection probability and predict what the audio sample label was.

3.3.3: Convolutional Neural Network (CNN)

In section 3 of Figure 4, we set up a model to train our fall detection sensor. The classification model chosen is based on the deep learning machine model. The base of this model uses a CNN (Convolutional neural network). This model type is well known for its classification abilities when used for image-based analysis. The CNN offers multiple feature extractions and classification components within the model. This model's ability for image feature extraction was the reason this model was chosen over other ML models. Using a sequential mode, starting with a simple model, the architecture comprises several convolution layers. These layers are all combined and stacked on top of the final dense layers. The overall model matches the number of possible classifications detected. The model will contain several features for its detection, allowing it to differentiate between other sounds to be sure it did not detect a fall.

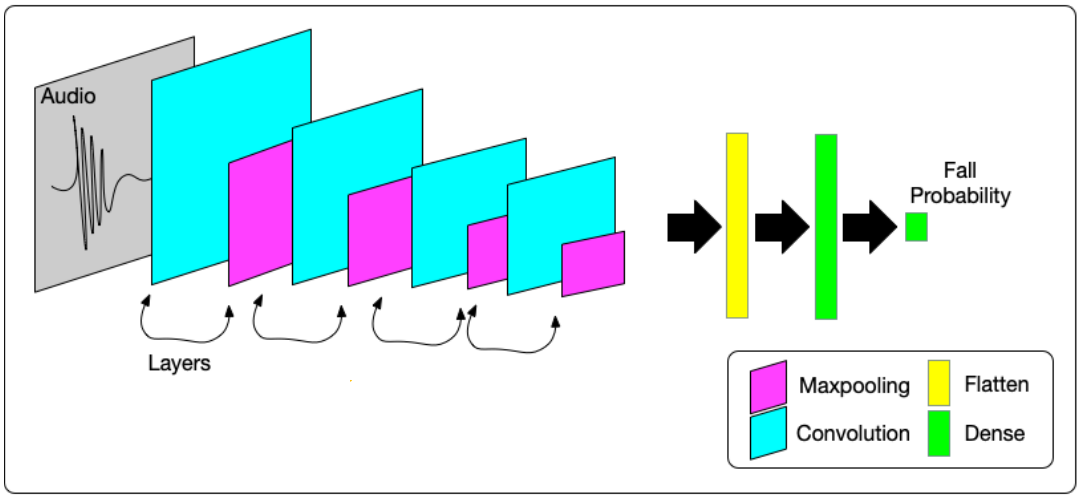


Figure 11: Fall Detection CNN Model

3.3.4: CNN System Layers

In this research, various models were tested with the pre-processed MFCC images from the A3Fall dataset. The models all use standard neural network components and variable convolution layers. We will detail each layer and the relevant information used to build the layer in the following subsections. The multi-layer CNN model will consist of the following structure:

Convolution Layer

Convolutional layers closest to the input typically learn fewer features, where ones closer to the output learn more features. The number of filters selected here is dependent on the complexity of our dataset and the depth of your neural network. We used standard settings to start with [32, 64, 128] for three layers and increasing to [256, 512, 1024]. Here we will begin with 32 and use 128 for the additional hidden later layers.

Padding

Padding on all layers has been, and the padding technique we use here is the padding = 'same.' This makes the input zero-padded so that the convolution output can be the same size as the input.

Kernel Initializer

The kernel initializer sets the initialization method used to initialize all values in the Conv2D class before training. Keras uses glorot_uniform as its default, which is Xavier Glorot uniform initialization and is suitable for most CNNs. We have used the he_normal initializer, as it's better suited for deeper networks, such as VGGNet. The he_normal initializer uses the MSRA initialization method for its initialization.

Kernel Size

Kernel size specifies the size of the convolutional filter in pixels. The kernel size is a matrix, with its height and width smaller than the data to be convoluted. It is also known as a convolution matrix or convolution mask. The kernel slides across the input data's height and width, and the dot product of the kernel and the image are computed at every spatial position. The length by which the kernel slides is known as the stride length.

In the model that we have used here, the kernel is of size 5X5, and the stride length is 1. The CNN architecture sometimes determines filter size example, VGGNet exclusively uses (3, 3) filters. Using the Keras recommend 5×5 filter for learning more extensive features, the training worked better on our A3fall dataset.

Activation Function

The activation function is the last component of the convolutional layer to increase the non-linearity in the output. The activation method options are typically based on the ReLU or Tanh function. When the input data is convolved with a kernel size, the new convolved feature of n-size is created to which the selected activation function is applied to get the output. In our models, we used the ReLu method.

Pooling Layer

The pooling layer is used to reduce the size of the input image. In a convolutional neural network, a convolutional layer is usually followed by a pooling layer. Pooling here is added to speed up the computation and detected more robust features. We used the max-pooling option for the pooling layers of our models. In max-pooling each part of a feature map calculates the maximum value and creates a reduced map. Average pooling is another method that finds the average rather than the maximum value function.

Fully Connected Layer

The last layer in our model is our fully connected one. Fully connected layers are typically used at the end of convolutional neural networks. The features map produced by the last layer is flattened to a long vector. This vector is then fed to a fully connected layer to capture all of the complex relationships between the high-level features. The output of the final layers is a one-dimensional feature vector.

3.4: TPU Sound Processing

When the device receives an audio sample, the system model will determine if it contains one of the pre-programmed models creating a corresponding classification score. This will permit the model to make the best accuracy for any model programmed event to detect and identify that an event has occurred. The system will then use the audio confirmation component and provide a positive feedback loop into the system as a validation of the detected sound. This permits the model to be continually learning and updating its model. The audio will be captured through the live sensor. Audio sensors can contain several different microphones in multiple locations to offer coverage in varying environments. This provides a rich set of samples for the system to process. The preprocessor breaks down the sample into a vector array of amplitude values. These values are then processed through a set of algorithms. One of the algorithms used in this model is the Fourier transformation function, turning the audio stream into its frequency components. Using a python package for music and audio processing will allow us to load the audio samples into a python system for analysis and manipulation. Feature extraction is performed by creating a visual representation of each sample of the audio. This allows the system to identify features for classification in the same method used to classify images. Spectrograms are one of the techniques used for visualizing the spectrum of frequencies, and they vary during a short period. Creating images using the Fourier transform technique and adding a similar process called the Mel-Frequency Cepstral Coefficients (MFCC) provides a feature extraction for training the ML model. MFCC uses a quasi-logarithmic spaced frequency scale that is closer to the human auditory system processing method.

3.5: Audio Data Capture

The main component of the system is the audio samples captured by a microphone. The system is designed around a Single Board Computer (SoC) with an SPI connected MEMS Audio sensor. The SoC uses an ARM architecture CPU and contains a built-in Bluetooth low energy stack for communication. The components are shown here in Figure 3.2 as individual blocks in the overall design.

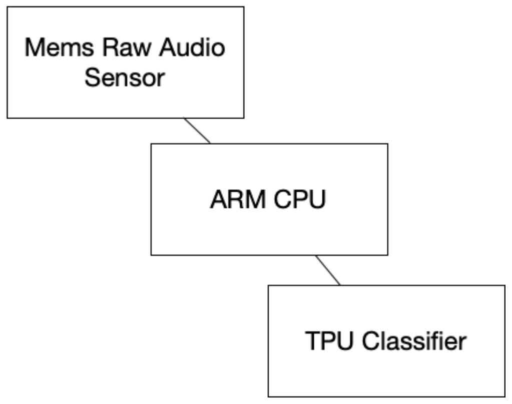


Figure 12: Overall Components of the Wireless Audio Sensor

The ARM CPU monitors the MEMS sensor's data, capturing small chunks of data and transmitting them back to the base station where the hardware tensor cores process the raw data through the machine model for classification.

3.5.1: Mems microphone

The ARM CPU monitors the MEMS microphone on the Arduino. The MEMS microphone sensor is directly attached to the CPU through an SPI interface. This allows the CPU to monitor and capture audio data in real-time. The captured data is processed into small packets and transmitted over Bluetooth low energy to the base station for the TPU to perform pre-processing and, finally, signal classification using the (CNN) Machine Learning model.

3.6: Multi-Sensor Deployment

The central hub, labeled as the TPU Server in Figure 13 (below), is the NVIDIA Jetson. The Jetson contains its tensor core processing units. These cores are used to help retrain the model and increase the speed of our prediction model in the systems edge processing power. The hub can also provide other (Non-Audio) sensors that may be used for increased fall accuracy—for example, the use of a vibration sensor or floor sensor. Using multiple sensors can expand the prediction system's capabilities and improve it beyond the simple audio processing.

Cloud deployment is possible with the additional connection from the TPU Server to a cloud server platform. Using a cloud deployment, the system can update and send alerts to an outside emergency system. Cloud deployment is optional and is not required for the system to work locally.

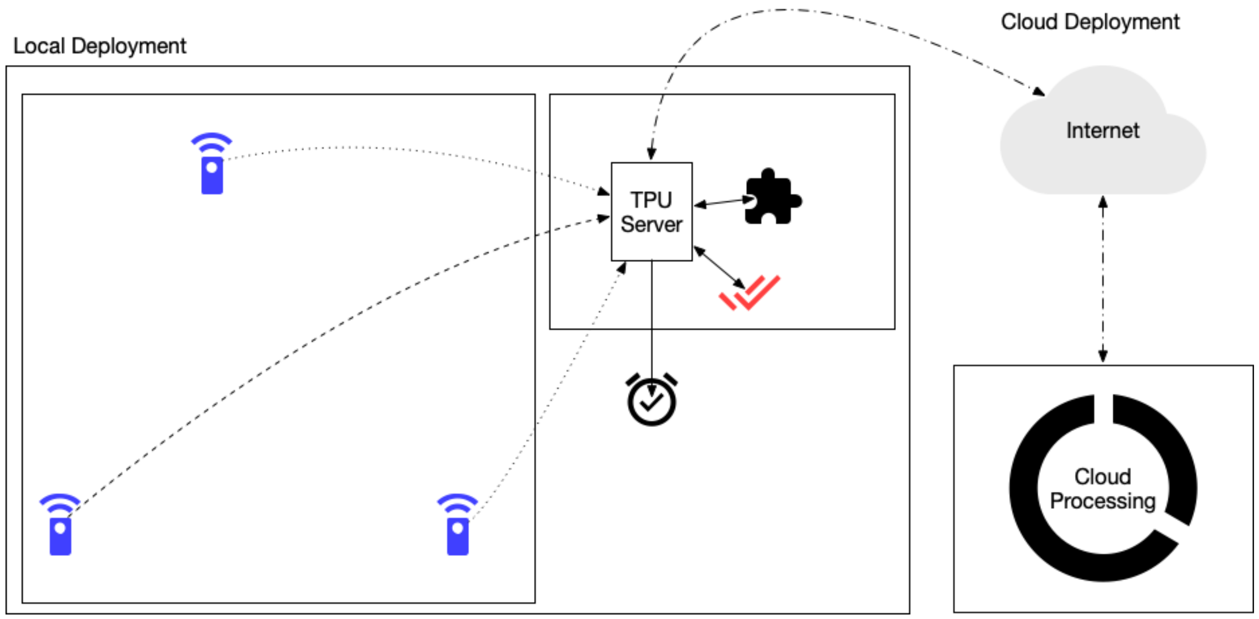


Figure 13: Multi-Sensor Deployment

During the audio sampling operations, the sensors create a probability matrix used by the central hub to determine if an event has occurred. Each sensor will continuously report its probability matrix while in operation. The central hub will process the information returned by these sensors and use an electoral type voting system to make a final decision for its prediction. The activation will be represented as a probability, and not all sensors are expected to return the same values. Sensor positions in various rooms will also help to map out where an event occurred.

The hub will use its hardware tensor cores for model updates and advanced learning while deployed. These updates are used to improve the CNN model over time, adding to the classification accuracy. In one experiment, the real human falls were separated from the simulated ones. This experiment was used to train the model only on simulated falls, and all tests were performed on the actual human falls. This experiment showed how the system could detect and classify events without recording the actual event that is being monitored.

3.7: Training and Testing the Network

The implementation of the testing and training was performed on the A3fall-v2.0 (used in previous works by D. Droghini et al. [7]), using hardware tensor cores GPU-enabled Python 3.7 instance. The model built here-in was build using Keras 2.3.1 and demonstrates how the network architecture will affect the classification of highly similar sound images.

The Data conversion process begins by processing functions that convert the .wav files into images in .jpg format. In this process, the extraction of the audio time-series data from each .wav file is performed. These files are loaded and then plotted into a spectrogram of the data. This data is then saved as a corresponding image. Audio samples are loaded into memory and converted into a floating-point value in the range of 0.01 to 1. This is required for faster ML classification in the model's analysis. The Convolutional Neural Network model was built utilizing the RMSProp optimizer.

3.8: Software Tools

The implementation was performed on the A3fall-v2.0, using hardware tensor cores on a GPU-enabled Python 3.7 instance. The model was build using Keras 2.3.1. Use of the Python library Librosa's was used during the pre-processing of the audio data.

Audio samples were loaded from the A3fall-v2.0 dataset using the .csv catalog file. Audio samples were processed and then stored as MFCC images in separate directories for the model training.

CHAPTER FOUR

RESULTS AND FINDINGS

4.1: Introduction

In this chapter we present the results of the research in two steps. First, we present the training and testing of the selected neural network models. We discuss the classification reports and confusion matrices of the model variants. Then, we present the results of deployment and testing the model in the Jetson Nano Development Kit.

4.2: Training and Test Data Set

Training Data was based on 10,000 audio events of varying length segments. Each segment was between under 2 seconds in length. The raw audio data was processed into spectrograms and Mel-frequency cepstral coefficients (MFCC). Training Data has 8000 labels corresponding to mic type, room location, and the added noise that corresponds to each of the images. Testing Data contains 2000 images of spectrograms; each image is 700x700 pixels. Testing Data has 2000 labels that correspond to each of the spectrogram images. Testing data classification is split between 49% falls and 51%, not a fall.

During training, the image data was normalized between 0.01 and 1. The process helps the CNN model add small weights to input values and prevent the 0 value from not being calculated. This modification to the pre-processing stage offered an increase in accuracy for the model.

Using the A3fall-2.0 dataset, the data was split into various categories based on each sound classification. The dataset was divided into train and test sets for validation of the final output from the system. During the initial data pre-processing, it was discovered that the choice of an appropriate filter and audio processing had a profound effect on the final model and its success or failure.

Simple spectrum analysis was not sufficient to generate a useable and recognizable model for classification. We expanded on the required pre-processing and added the additional use of the Mel-spectrum coefficients (MFCC).

Switching between different resolutions for our dataset images had a minimizing effect when the images were rendered at or above 700x700px. Larger sample images did not offer any significant improvement for the image classification.

The classification results were directly affected by different conversion techniques used to convert the audio from a time-frequency domain into an image. Early initial processing was based on splitting the train and test data into an 80/20 randomized fashion. The final experiments used the mimicking doll audio as the training data, and the real human falls for the testing data. Without using the actual human falls from the dataset, the results provide a more realistic approach to testing in an environment where the CNN would never see training data for a real fall event.

4.3: TRAINING THE MODEL

In this research, we tried VGGNet, LeNet, and ConvNet architectures. With the initial runs, we found that LeNet and ConvNet performed very poorly in terms of model accuracy and classification. These two models did not converge over the training process. On the other hand, initial results from VGGNet architecture proved to be more promising. Therefore we continued with this architecture.

As done in VGGNet, the process of stacking convolution layers was used to increase the filters' complexity for refined feature extraction. We created multiple variants of the base VGGNet model, varying CNN's hyperparameters. The purpose was to find an optimal combination of layers, filters, and parameters. The summary of model parameters is presented in Table 2. The variants are labeled according to the number of convolutional layers. For example, M2B means the model contains two convolutional layers. A, B, and C imply variation in input size or number of hidden layers.

Table 2: Summary of Models Parameters

CNN Layers Variants	Input Size	Hidden Layer Filters	Dense
M1	270x270x3	NA	128
M2A	270x270x3	64	128
M2B	270x270x3	128	128
M2C	70x70x3	128	128
M3A	270x270x3	64,128	256
M3B	270x270x3	64,128	128
M4	270x270x3	64,128,128	256
M5	270x270x3	64,128,128 128	512

4.4: TESTING THE MODELS

The confusion matrix comparison below shows how each model variant performed in detection and accuracy. The training of the models uses validation loss as the method to quantify the accuracy of the model. This method allows the neural network to favor false positives over false negatives. False positives can be confirmed through verbal communication, where false negatives would mean falls were not being detected.

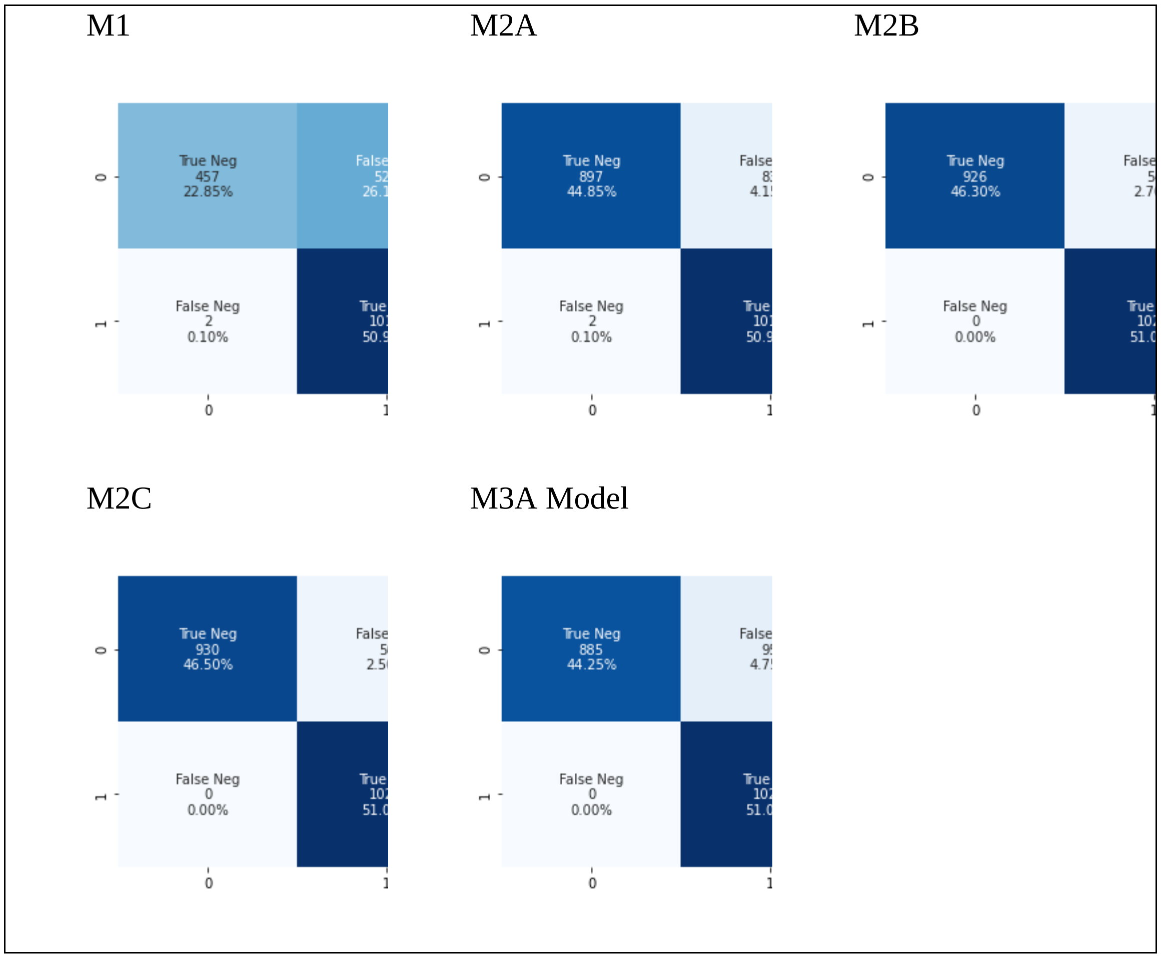


Figure 14: Comparison of Confusion Matrices

As can be observed in Figure 14 above, M2C and M3A performs better than our other models when compared in terms of accuracy. However, the M2C model is based on only 2,388,673 parameters compared to 36,795,073 in M3A.

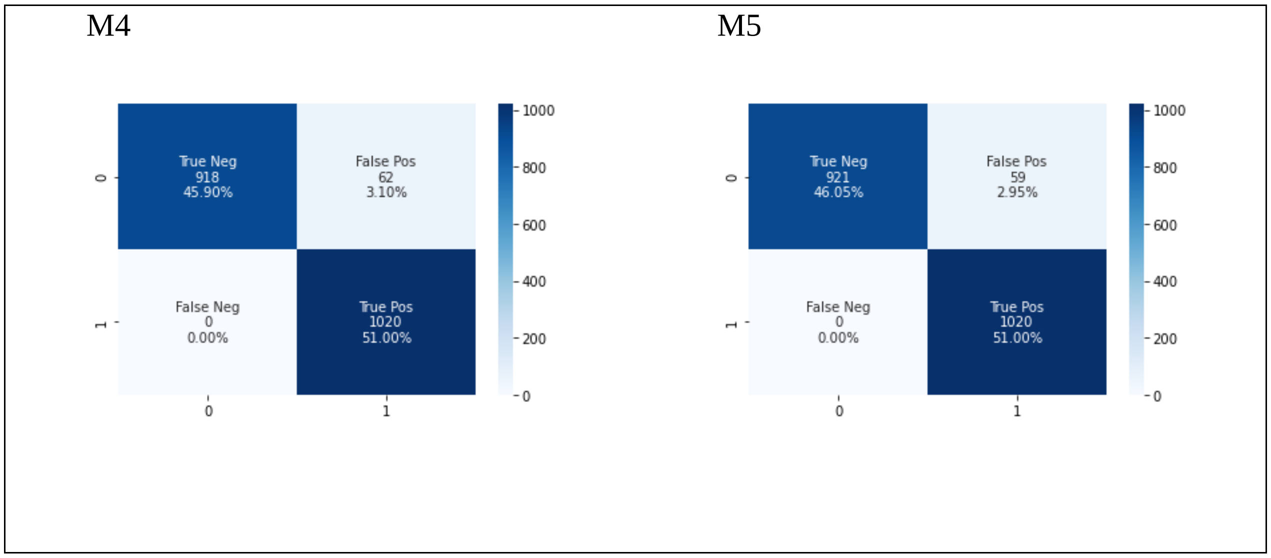


Figure 15: Comparison of Confusion Matrices: M4 and M5

M4 and M5 perform slightly better than our M2, and M3 models do. M4 offers a false positive rate of 3.10%, where M5 is only 2.95%. Adjustment of the probability threshold can bring these models to a higher accuracy over M2C. However, the slight improvement comes at a much higher model complexity. Further research into using these models will not continue as these models are far too complex for our needs. Our primary focus here is a final hardware deployment, and a simpler model is preferred for its computational efficiency.

Table 3 presents a summary of classification reports for all the model variants shown in Table 2. In subsequent sections, we will discuss the details of each model's measured performance.

Table 3: Summary Classification Report Of All Models

CNN Layers Variants	Training Accuracy	Testing Accuracy	F1-Score	Precision	Recall
M1	0.96	0.74	0.64	1.00	0.47
M2A	0.98	0.94	0.95	1.00	0.90
M2B	0.99	0.94	0.97	1.00	0.94
M2C	0.99	0.97	0.97	1.00	0.94
M3A	0.99	0.964	0.95	1.00	0.90
M3B	0.99	0.967	0.97	1.00	0.94
M4	0.99	0.967	0.97	1.00	0.94
M5	0.99	0.968	0.97	1.00	0.94

During training, the model could achieve a 97% accuracy in all variants with 2, 3, and 4 convolutional layers. While working through multiple epochs, the training can take more than 100 to improve the model's accuracy to an acceptable level. The model's complexity was significant as the more straightforward the model, the easier it will be to transition the model onto the IoT device for edge processing. The precision and recall of the model were necessarily the most critical metric within our testing results. With a precision of 1 on all models and a maximum recall of 0.94, the simplest model that provides the highest accuracy is the M2C variant.

The M2C variant uses a reduced image size in its input processing. The initial layer's input size is 70x70x3, and the model only uses one hidden convolution layer with 128 filters. This model was comparable to the 3, 4, and 5 layer versions and produced very similar results. In the confusion matrix Figure 4.3, we can see how close each of these models performed based on their testing data classification.

In these subsequent sections, we will compare the M2B/C and M3A. These are our candidates for hardware implementation.

Comparing M2B and M2C

The two-layer CNN M2B and M2C only vary based on the initial image sized input. In the M2B model, a larger 270x270x3 image was used for training. In the M2C model, a smaller 70x70x3 image size was used. Below is the Keras model configuration in Python for both models M2B and M2C. While the M2B architecture, M2C models are identical, the size of input/outputs is smaller in M2C and has fewer filters to configure.

Table 4: M2B, M2C Layer CNN Architecture

Operation Layer	Filters	Filter Size	Kernel Size	M2B Output	M2C Output
MFCC Image	-	-	-	270 x 270 x 3	70 x 70 x 3
2D Convolution Layer Activation	Convolution ReLU	32	5 x 5	270 x 270 x 32	70 x 70 x 32
Pooling Layer	Max pool-ing	1	2 x 2	135 x 135 x 32	35 x 35 x 32
2D Convolution Layer Activation	Convolution ReLU	64 -	1 x 1 -	135 x 135 x 64	35 x 35 x 64
Pooling Layer	Max pool-ing	1	2 x 2	67 x 67 x 64	17 x 17 x 64
Flatten	-	-	-	287296	18496
Dense	ReLU	128	-	128	128
Dense	sigmoid	1		1	1
Total params: 36,795,073 Trainable params: 36,795,073 Non-trainable params: 0					2,388,673 2,388,673 0

Table 4 above shows the parameters for the comparison between variant B and C. In model B, there are 36,795,073 trainable parameters, where model C only has 2,388,673. Both models required over 100 epochs for training these models. The historical details of the training process are outlined in Figure 16 below.

During training, both M2B and M2C achieved a high level of accuracy with only 100 epochs. M2B required 100 epochs to finish its training, and M2C needed 120 epochs. The training of these models used an early stopping method based on the validation loss.

Table 4, the classification report of M2B, and M2C shows that the model's recall and precision. These two models have a precision of 1.00, M2B has a recall of 0.90, and M2C has a higher recall of 0.95. The average accuracy of the M2C model is 0.98 for precision and 0.97 for recall. This model has a less complicated structure and offers the best accuracy for its complexity.

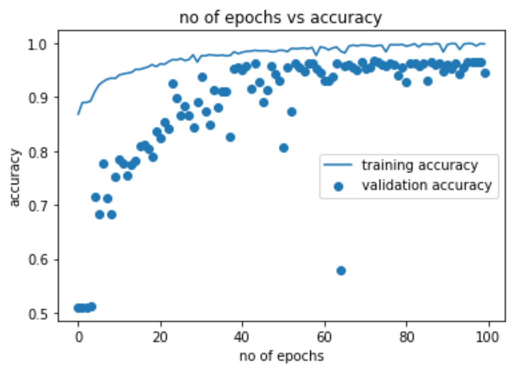
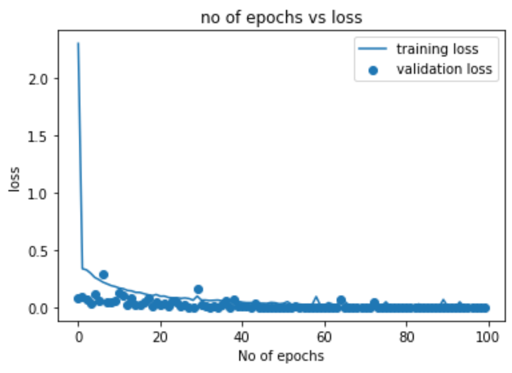
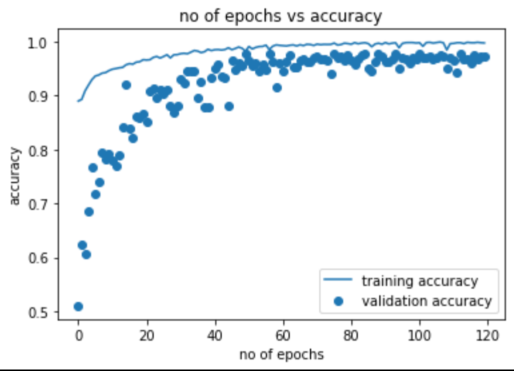
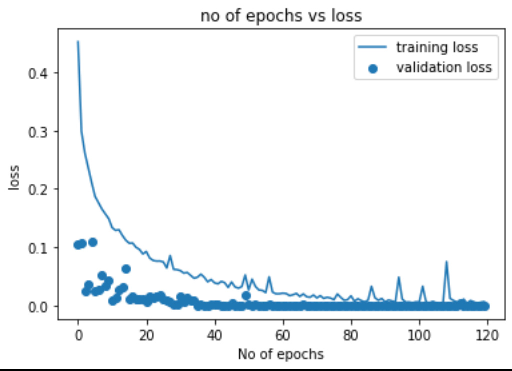
Prediction accuracy of CNN training	Loss value of CNN during training
M2B 	M2B 
M2C 	M2C 

Figure 16: M2B/M2C Training and Validation Loss on CNN

Table 5: Classification Report M2B

	Precision	Recall	F1-score	Support
Fall	1.00	0.90	0.95	980
No fall	0.91	1.00	0.96	1020
Accuracy			0.95	2000
Macro avg	0.96	0.95	0.95	2000
Weighted avg	0.96	0.95	0.95	2000

Table 6: Classification Report M2C

	Precision	Recall	F1-score	Support
Fall	1.00	0.95	0.97	980
No fall	0.95	1.00	0.98	1020
Accuracy			0.97	2000
Macro avg	0.98	0.97	0.97	2000
Weighted avg	0.98	0.97	0.97	2000

Comparing M2C and M3A

Finally, we compare the best M2 model with the M3A model. Here the M3A model offers higher accuracy in its false-negative detection. However, the model is far more complicated than M2C. In the table below, the outline of the M3A model is shown. Here we compare M2C and M3A.

Table 7: M3A Layer CNN Architecture

Operation Layer	Filters	Filter Size	Kernel Size	M3A Output	M2C Output
MFCC Image	-	-	-	270 x 270 x 3	70 x 70 x 3
2D Convolution Layer Activation	Convolution ReLU	32	5 x 5	270 x 270 x 32	70 x 70 x 32
Pooling Layer	70 x 70 x 32	1	2 x 2	135 x 135 x 32	35 x 35 x 32
2D Convolution Layer Activation	35 x 35 x 32	64 -	1 x 1 -	135 x 135 x 64	35 x 35 x 64
Pooling Layer	35 x 35 x 64	1	2 x 2	67 x 67 x 64	17 x 17 x 64
2D Convolution Layer Activation	17 x 17 x 64	128 -	1 x 1 -	67 x 67 x 128	-
Pooling Layer	18496	1	2 x 2	33 x 33 x 128	-
Flatten	128	-	-	139392	18496
Dense	1	256	-	256	128
Dense	70 x 70 x 32	1		1	1
Total params: 35,779,649 Trainable params: 35,779,649 Non-trainable params: 0					2,388,673 2,388,673 0

The M2C model, listed in Table 7, has fewer convolution layers than M3A. The model also uses a smaller input size of only 70x70, down from 270x270 in M3A. These two models set these differences apart as the model's complexity of M3A is much higher than M2C with little improvement in the usable accuracy.

Figure 17 shows the comparison between the training and testing of M3A and M2A. Both models were trained using over 100 epochs. M2C was always consistently higher in its validation accuracy than M3A. During training, M2C validation and training loss had still not settled, where M3A the graph shows the two converging after approximately 40 epochs. See Figure 17 below.

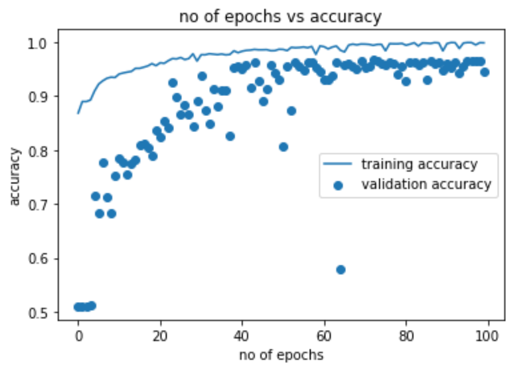
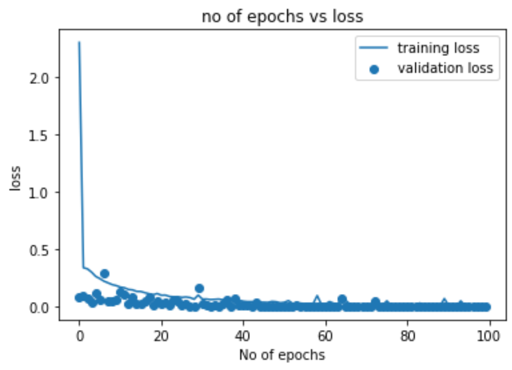
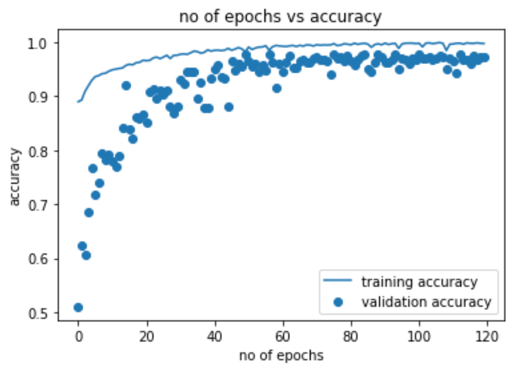
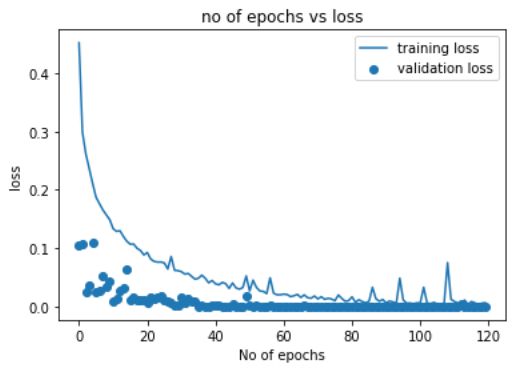
Prediction accuracy of CNN training	Loss value of CNN during training
M3A 	M3A 
M2C 	M2C 

Figure 17: M3A/M2C Training and Validation Loss History

The following classification report in Table 8 shows the precision, recall, and f1 score of the M3A model. The report averages include the macro average (averaging the unweighted mean per label) and weighted average (averaging the support-weighted mean per label). Compared to M2C from Table 7, the M3A model's performance is on par with the M2C model.

Table 8: M3A Classification Report

	Precision	Recall	F1-score	Support
Fall	1.00	0.90	0.95	980
No fall	0.91	1.00	0.96	1020
Accuracy			0.95	2000
Macro avg	0.96	0.95	0.95	2000
Weighted avg	0.96	0.95	0.95	2000

4.5: Hardware Implementation

Transferring the model to the NVIDIA Jetson was achieved by copying the pre-built model over to the Jetson's file system. Using test data saved from the training/test split, the model was once again tested on the Jetson hardware. The testing resulted in the same accuracy as was observed on the training computer setup.

4.5.1: Hardware Implementation – Transfer and Run

Running the Model on the NVIDIA Jetson using Keras-GPU enabled Python libraries allowed the model to perform similarly as when running the training computer model. The trained model ran with no significant measure of CPU usage while testing the models on the Jetson hardware. Data transfer from disk to memory was the most intensive task recorded during testing.

In this proof of concept, we used sample audio files from the same dataset (A3Fall), which this model did not see before. We are simulating the capture of sound audio using the original dataset .wav files.

During the Jetson testing, the model could compute the predictions for our 2000 sample audio files in seconds. Measuring the Jetson's performance was an issue as there is a limitation on storing and retrieving the data from the file system. The Jetson uses SD cards, and the transfer rate may be an issue. The Jetson ARM CPU and Tensor Core Processors did not record any measurable load while operating on the test set.

4.6: Conclusion

In this research, we implemented different machine learning models to realize fall detection using audio samples. First, we extracted the audio dataset features using filters along with the Mel-frequency spectrum cepstral coefficients (MFCC). Each audio sample was converted into a representation image. Based on the image audio representation, we used five different convolutional neural networks to contrast and compare the results. Using varying hyperparameters, we found the best computational model by balancing the model's complexity with accuracy.

In training our model variants, the accuracy increased when adding inner layers with higher filter numbers. The higher filter numbers were located towards the end of the model. Early layer filters in the model targeted the features that are common to all the MFCC spectrogram images. When we reach the later layer(s) together with their convolved output maps, the MFCC spectrograms are similar enough that a higher number of complex filters helps to distinguish between them.

Hardware deployment on the Jetson was successful, and using the Tensor Processing Units; the hardware had no trouble in processing the testing dataset.

CHAPTER FIVE: CONCLUSION AND FUTURE RESEARCH

5.1: Conclusions

We reviewed various methods of sensing in environments where fall detection is a concern.   Video monitoring may be a reliable detection system for falls in the home. With the mass production of security cameras and in-home surveillance, the hardware has become very inexpensive. The video detection model also does not have the same drawbacks as the wearable sensors. Video detection offers little to no intervention by the user and will allow seniors to maintain any system aspect. The main problems with camera surveillance are the constant monitoring and uncomfortably of being watched. Video-based detection is optimal for its accuracy and low cost; however, it's very intrusive to the Senior [1]. There are also placement problems with cameras. Each camera's coverage area will be within its field of view, and cameras placed in bedrooms or bathrooms pose other ethically charged discussions. Using a group of low-frequency audio sensors in place of large-scale infrastructure investment on structural changes allows the deployment of this system in many areas. The additional installation and maintenance of an extensive sensor network would be too cumbersome and, overall, costly.

Using sensors, such as audio detection can detect events before other sensing forms (motion and vibration). These features could predict the occurrence of a fall based on our CNN model. The trained model has shown that a camera safety system can be replaced with a sensor using only audio signals. Capturing the data using a specialized audio sensor increases the overall accuracy and can sense details for locations where a camera-based system would not have a clear vision. Detection using audio offers a complete area coverage. Individuals will not need to have cameras installed throughout their homes. Specialized floor and room renovations do not need to be performed to achieve a reliable method for sensing falls in the house. The proposed system approach addresses many individual issues with current technologies with the addition of personal privacy and security.

5.2: Limitations

Few research that is done with similar audio based fall detection. As most of the research is done using video systems direct comparison of performance with our research proved to be difficult. While the training datasets exist for video-based systems, the sound channels are not available to train our model for performance comparison.

It may be possible to deploy both video and audio detection systems and run experiments. However, that will require extensive experimental setup and volunteer participation which is beyond the scope of this research.

5.3: Future works

Our research has shown that accurate fall detection using audio is feasible. The testing of our final model into a TinyML model and run on a Jetson Nano Developer Kit. Using a low-cost device and running the model exclusively on the Jetson Kit would have a tremendous cost reduction.

More research can be performed into an acoustic enclosure's designs to hold the Jetson Kit and microphone. The design needs to be based on a box bandpass enclosure to maximize the low frequencies and high-frequency filtering. This would further the design into a floor sensor with all the elements of the detection system that are self-contained in a single unit, expanding on the system's initial probability nature, implementing a voting group electoral classification system.

Cloud-based uploads and learning for model evolution would offload all audio processing to the cloud. Samples would be uploaded to the cloud TPU and processed to update existing deployed units.

Additional signal processing can be performed on the signals captured to filter out any ambient white noise or burst noise that the sensors may pickup from surrounding mechanical or electrical sources.

REFERENCES

[1] CBC, "Use of surveillance tech to monitor Seniors," 2014. [Online]. Available: https://www.cbc.ca/news/technology/use-of-surveillance-tech-to-monitor-seniors-at-home-on-rise-1.2535677.

[2] Statistics Canada, "Understanding seniors' risk of falling and their perception of risk," 2009. [Online]. Available: https://www150.statcan.gc.ca/n1/pub/82-624-x/2014001/article/14010-eng.htm.

[3] Telus Health, "Technology key to keeping Canadian seniors healthy at home," 13 November 2018. [Online]. Available: https://www.telushealth.co/item/technology-key-keeping-canadian-seniors-healthy-home/.

[4] Hanba, Curtis, Gupta, Amar, Svider, F. Peter, Folbe, J. Adam, Eloy, A. Jean, Zuliani, F. Giancarlo, Carron and A. Michael, "Forgetful but not forgotten: Bathroom-related craniofacial trauma among the elderly," *The Laryngoscope*, vol. 127, no. 4, p. 820, 2017.

[5] A. Bourke, J. O'Brien and G. Lyons, "Evaluation of a threshold-based tri-axial accelerometer fall detection algorithm.," *Gait & Posture 26.2*, pp. 194-199, 2007.

[6] R. Bajcsy, J. Chen, K. Kwong, D. Chang and J. Luk, "Wearable sensors for reliable fall detection.," *Engineering in Medicine and Biology Society*, pp. 3551-3554, 2005.

[7] Apple Computer, "Apple Watch - Battery Apple(CA)," 2019. [Online]. Available: https://www.apple.com/ca/watch/battery/. [Accessed 2019].

[8] D. Droghini, F. Vesperini, E. Principi, S. Squartini and F. Piazza, Proceedings of the International Conference on Pattern Recognition and Artificial Intelligence, New York, NY, USA: Association for Computing Machinery, 2018.

[9] C. Lehan, K. Xiangbo, T. Hiroyuki and M. Lin, "Multiple States Fall Detection System for Senior Citizens," in *2019 International Conference on Advanced Mechatronic Systems (ICAMechS)*, 2019, pp. 169-174.

[10] A. Garofalo, M. Rusci, F. Conti, D. Rossi and L. Benini, "PULP-NN: accelerating quantized neural networks on parallel ultra-low-power RISC-V processors," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences,* p. 20190155, 2019.

[11] H. Sajid, Z. E. Senol and J. H. Park, "Monitoring User Activities in Smart Home Environments," *Information Systems Frontiers,* pp. 539-549, 2009.

[12] Y. Zhiqi, "Gesture recognition based on improved VGGNET convolutional neural network IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC)," in *2020 IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC)*, 2020, pp. 1736-1739.

[13] D. Sun, Y. Chen, J. Liu, Y. Li and R. Ma, "Digital Signal Modulation Recognition Algorithm Based on VGGNet Model," in *2019 IEEE 5th International Conference on Computer and Communications (ICCC)*, 2019, pp. 1575-1579.

[14] G. Zumpano and M. Meo, "On the optimal sensor placement techniques for a bridge structure," *Engineering Structures,* vol. 27, no. 10, pp. 1488 - 1497, 2005.

[15] L. Hårvik, C. Madshus and B. Bessason, "Prediction model for low frequency vibration from high speed railways on soft ground," *Journal of Sound and Vibration,* pp. 195 - 203, 1996.

[16] N. Ball and A. Baydar, "A Comparative Study of Acoustic and Vibration Signals in Detection of Gear Failures using Wigner-Ville Distribution," *Mechanical Systems and Signal Processing,* pp. 1091 - 1107, 2001.

[17] A. Ball and N. Baydar, "Detection of gear failures via vibration and acoustic signals using wavelet transform," *Mechanical Systems and Signal Processing,* pp. 787-804, 2003.

[18] C. Fischer, "Feedback on household electricity consumption: A tool for saving energy?," *Energy Efficiency*, pp. 79-104, 2008.

[19] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE,* pp. 257-286, 1989.

[20] T. Ueno, F. Sano, O. Saeki and K. Tsuji, "Effectiveness of an energy-consumption information system on energy savings in residential houses based on monitored data," *Applied Energy,* pp. 166-183, 2006.

[21] W. Kempton and L. Layne, "The Consumer's Energy Analysis Environment," *Energy Policy 22(10),* pp. 857-866, 1994.

[22] H. Matthews, L. Soibelman, M. Berges and E. Goldman, "Automatically Disaggregating the Total Electrical Load in Residential Buildings: a Profile of the Required Solution," *Intelligent Computing in Engineering, Plymouth,* 2014.

[23] L. Leeb, K. Norford and B. Steven, "Non-intrusive electrical load monitoring in commercial buildings based on steady-state and transient load-detection algorithms," *Energy and Buildings*, pp. 51-64, 1996.

[24] H. Jiawei, K. Hyungsul, M. Manish, A. Martin and L. Geoff, "Unsupervised Disaggregation of Low Frequency Power Measurements," in *Proceedings of the 2011 SIAM International Conference on Data Mining*, 2011, pp. 747-758.

[25] A. Hampapur, L. Brown, J. Connell, A. Ekin, N. Haas, M. Lu, H. Merkl and S. Pankanti, "Smart video surveillance: exploring the concept of multiscale spatiotemporal tracking," *IEEE signal processing magazine*, pp. 38-51, 2005.

[26] J. Dan-Ning , L. Lie, Z. Hong-Jiang, T. Jian-Hua and C. Lian-Hong, "Music type classification by spectral contrast feature," in *Proceedings. IEEE International Conference on Multimedia and Expo,* 2002, pp. 113-116.

[27] M. Alwan, Rajendran, S. Kell and D. Mack, "A smart and passive floorvibration based fall detector for elderly.," *Information and Communication Technologies,* pp. 1003-1007, 2006.

APPENDIX