GRADUATE THESIS APPROVAL FORM AND SIGNATURE PAGE

Instructions: This form must be completed by all doctoral students with a thesis requirement. This form MUST be included as page 1 of your thesis via electronic submission to ProQuest.

Thesis Title: On the Real-World Interactivity Potential of Minimalistic Knitted Sensors at the Intersection of Artificial Intelligence and User Experience

Author's Name: Denisa Qori McDonald

Submission Date: 01/04/2022

The signatures below certify that this thesis is complete and approved by the Examining Committee.

Role: Member Name: Dario Salvucci
Title: Professor
Department: Computing
Approved: Yes Date: 01/04/2022

Role: Chair Name: Ali Shokoufandeh
Title: Professor
Department: Computing
Approved: Yes Date: 01/04/2022

Role: Member Name: Genevieve Dion
Title: Professor
Department: College of Media Arts & Design
Approved: Yes Date: 01/05/2022

Role: Member Name: Erin Solovey
Title: Assistant Professor
Institution: Worcester Polytechnic Institute
Approved: Yes Date: 01/04/2022

Role: Member Name: Jeremy Johnson
Title: Professor
Department: Computing
Approved: Yes Date: 01/04/2022

On the Real-World Interactivity Potential of Minimalistic Knitted Sensors at the Intersection of Artificial Intelligence and User Experience

A Thesis Submitted by Denisa Qori McDonald

December 2021

Acknowledgment

I would like to thank everyone that has supported me through this experience. I am deeply grateful for your help and encouragement. In particular, I would like to thank my committee members starting with Dr. Ali Shokoufandeh my advisor and mentor, who has supported and promoted my work, and whose help has been fundamental to me reaching this day; Prof. Genevieve Dion, my other advisor, for enabling my research and for her insights and helpful comments; Dr. Erin Solovey who has supported me throughout my PhD and has been a true mentor; and Drs. Jeremy Johnson and Dario Salvucci for helping me finalize and distill this work. I would also like to thank my collaborators, particularly Rich, for being an integral part in this work, which was highly-collaborative. I would like to thank Shruti and Lev who helped bring this work to fruition as well, and everyone at the Center for Functional Fabrics for producing the sensors whenever it was necessary.

On a personal note, I would like to thank my mother-in-law, Mirna, for always encouraging my research career path. I would like to thank my sister, Egla, for always being by side and brightening my day; my parents, Merita and Bedri, for the outpour of love, support, and for serving as a constant inspiration. Finally, I would like to thank my husband, Andrew, for always being there for me, through the ups and downs, encouraging me, and helping me stay focused on my path.

Table of Contents

List of Figures

Abstract

On the Real-World Interactivity Potential of Minimalistic Knitted Sensors at the Intersection of Artificial Intelligence and User Experience
Denisa Qori McDonald
Ali Shokoufandeh, PhD and Genevieve Dion

Recent work has demonstrated the feasibility of producing knitted capacitive touch sensors through digital fabrication, which rely on a single conductive yarn and two external connections. This technique increases these sensors' robustness and usability, while shifting the complexity of enabling interactivity from the hardware to computational models. The application of algorithmic and artificial intelligence models to these novel pervasive technologies is key to unfolding their potential, particularly when real-world and user experience considerations are also included. To bring this technology closer to real-world use, this dissertation goes beyond previous work on coarse touch discrimination, to enable fine, accurate touch localization and complex gesture recognition on these low-profile knitted sensors. Deep learning and algorithmic models are presented to analyze noisy time–series signal data, which are able to capture the temporal behavior of the sensors and extract relevant local features. Furthermore, several user studies are conducted to train these models, demonstrate their generalizability with new users, and investigate their robustness when exposed to everyday use events. To start shaping the future of touch-sensitive fabric technology according to user expectations and everyday use scenarios, through a formative focus group study, users' views of these fabrics are also explored in different contexts. The contributions of this work set the foundations for creating pervasive interactive systems powered by artificial intelligence models that use minimalistically-designed knitted sensors as an input medium.

Chapter 1: Introduction

1.1 Overview and Motivation

The emergence and increasing relevance of machine learning unfolds previously inaccessible opportunities for real–world ubiquitous technologies. As computing advances toward becoming more human–centered and intuitive to use, interactive devices have evolved. Devices that are smaller in size, contain fewer hard electronic elements, and are seamlessly integrated into our environments have emerged. These advancements allow us humans to keep our perception of the world coherent, while enabling novel interactions. Weiser famously claimed: *"The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it."* [140]. Computing conceptualized this way aims to psychologically free the user from dedicating active attention to its working, to focus on further goals [43]. However, the increased usability potential, many times, comes at the expense of reducing information obtainable from such devices, due to their minimalistic designs. Additionally, the relevant information needs to be extracted from the noise, which has multiple, and sometimes not directly identifiable sources in the real world, where this technology needs to operate. Therefore, it is necessary to use more complex computational models in order to interpret the signals received as a response to human interaction with such minimalistic devices.

Neural networks, which are machine learning algorithms whose construction is inspired by the connections between neurons in our brain, aim to detect complex relationships in large datasets. They have been successfully used in many applications, especially in the fields of computer vision and natural language processing. Their applicability also extends, although to a smaller extent, to sensor–generated signals, part of which are minimalistic devices used for enabling HCI interactions. Much of the complexity and unexpected aspects of signals encountered during everyday use of small, lightweight devices, can be reduced in favor of more focused outcomes with the help of neural networks.

One of the main obstacles to effective adaptation of neural networks in the field is the fact that it is difficult and time–consuming to collect physiological data from users, resulting in smaller datasets, specific to conditions investigated for each experiment. Additionally, in the data collection process for creating interactive systems, the experimental design aspect is essential. So is ensuring the validity of the collected data, as well as any necessary pre–processing steps to represent the data in ways that are meaningful, before using it as an input to learning algorithms. Many times, depending on the study, it is also not possible to directly use external machine learning libraries on all the collected data, since it is necessary to account for the variation among subjects, or the similarity of the data from the same subject. Moreover, the unpredictability of real environments as reflected in the data should also be considered in this process, as should users' views and perceptions of novel technologies.

These considerations and others are part of the process of creating machine learning models that work with real datasets in this context. In order to successfully build real–time interactive systems that use signals generated by users' activity as an input, an interdisciplinary approach would be necessary, drawing information from multiple fields, such as machine learning, combinatorial algorithms, signal processing, experimental design, usability studies, software engineering, electrical engineering and more.

1.2 Focus Area

This work focuses on building computational models for interactive, minimally–designed, real–time systems, while also considering users' views and everyday use conditions. The aim of the algorithms and neural network architectures designed is to capture and interpret the relevant aspects of signals output from these systems, which would make them suitable for use by humans in the real world. This theoretical construct finds realization in the domain of functional fabrics. As a use case, one particular subclass of smart textiles is studied: knitted sensors relying on one conductive carbon–coated yarn and only 2 or 4 external electrical connections to receive user input on a 2–dimensional surface area. These previously introduced sensors [133], are produced by combining conductive yarn

with regular, non–conductive yarns using digital knitting (fig. 1.1). The number of connections from the fabric to outside hardware is kept to a minimum in order to enhance sensor usability and robustness, as well as simplify its construction for manufacturing ease.

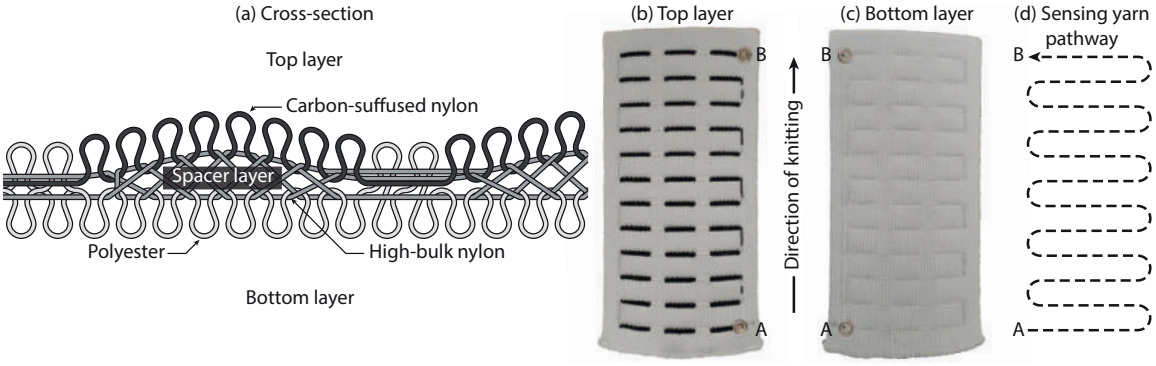


Figure 1.1: Weft–knitted touch sensor design and construction. Figure produced at CFF [3]. *(a)*: Illustration of the layering formed by digital flatbed weft knitting with conductive carbon–coated nylon yarn, and non–conductive polyester and high–bulk nylon yarns. *(b)*: Image of the top layer of a completed sensor. A and B denote the yarn endpoints, where external electrodes can be connected. Touch areas are shown in black. *(c)*: Image of the bottom layer of a completed sensor. There are no sensing elements on this layer. *(d)*: Illustration of the serpentine pathway of the carbon–suffused nylon created during the knitting process.

While this work focuses on a few selected sensor patterns to evaluate the potential that machine learning and algorithms unfold, those are not the only designs that can benefit from their power. In fact, the whole ecosystem of similarly produced sensors can. The sensor design process allows, among other aspects, the production of related patterns of different sizes, ensuring design scalability; flexibility in the number of connections to external hardware; rapid prototyping to test novel design patterns; and manufacturing–friendly production, since wiring is kept to a minimum and the design is repeatable. While such sensors are produced in an application–agnostic way and offer many possibilities, their full realization is thwarted without accurate localization abilities and real–world usability considerations. The application of machine learning unlocks their potential, by allowing interaction and fashion designers to decide on the best pattern for a particular sensor, only depending on the application needs, usability concerns, and machine knitting technicalities, unencumbered and unlimited by the location identification or gesture recognition accuracy of the signal that a particular pattern outputs. Therefore, this platform of knitted sensors provides a comprehensive environment for evaluating the effect of designing and applying novel neural network architectures and algorithms

to obtain information from the state or behavior of devices designed with few output channels.

The signal processing approach of these existing knitted sensors relies on differential capacitive sensing [133], which is compatible with the yarn routing pathway of the knitting machines. However, that approach alone does not disambiguate the location of touch accurately: it provides only coarse touch localization. The reduced information output from the system due to few connections requires more complex computational models to extract touch location information, the relevant signal from an HCI perspective. This provides an opportunity to use and develop more refined methods of representing the signals output from these minimalistic, easy–to–use, knitted sensors to detect location of touch and gestures.

My hypothesis is that it is possible to build ubiquitous, interactive systems using minimalistic knitted sensors as an input medium, and recognize through deterministic and stochastic models, human interaction events, such as location of touch and complex gestures. This solution would enable the creation of real-time systems that are designed with lightweight hardware and minimal wiring, in order to adapt to intuitive and comfortable real–world use by humans, without sacrificing accuracy and expressive power.

The central complexity of this work is designing novel computational models to represent the signals output form the system, such that location and gesture identification information is extracted. These models primarily rely on long–short term memory (LSTM) networks and convolutional neural networks (CNN). The former ones are designed to capture temporal dependencies in sequential data, signals in this case, and the latter focuses on spatial dependencies in data. The process of creating such systems relies mainly on machine learning and algorithmic developments, but also on quantitative and qualitative user studies, and signal processing techniques. Next, the challenges of this investigation are explored more thoroughly.

1.3 Challenges for Real–World Interactivity

The central computational problem related to this knitted sensor technology is *the reduced amount of information* that is output from the system. To make the sensor design more usable, the man-

ufacturing more streamlined through digital machines with minimal human intervention, and the integration with other sensors or devices easier, the external wiring is kept to a minimum. This contrasts a lot of existing textile sensors that require connecting many thin wires to the fabric components, or which have several layers of electrodes. This hardware design choice transfers the responsibility for accurate touch location and gesture recognition to computational models, as the lightly processed signal response the circuit outputs [134] is relatively vague regarding these fundamental functionalities, which would serve as the basis for building any interactive applications. The effect of this problem is amplified by *the challenges of textile behaviour*. In order for a textile sensor to be truly functional, it has to retain the traits of flexibility, stretchability and dynamic motion [40] without interfering with the application accuracy.

Another challenge related to signal interpretation, is *data variability*, which is prevalent in physiological signal data. This technology relies on users touching on the interactive areas of the knitted sensors, creating a capacitor between the human skin and the conductive knitted component. There are several factors potentially affecting the signal, such as the variability in users' skin conductance, different finger placements, different body poses, performing gestures while in motion, and many more. For these sensors to intuitively work in the real world, the signal interpretation should be user–independent, not relying on individual calibration. Moreover, physiological data is not straightforward to process, and often requires designing specialized data processing, analysis, and storage tools. Related to the issue above is that of the *data shortage* regarding this specific technology, especially problematic when using deep learning models, which typically rely on a large amount of data to produce accurate results. While recent years have seen a massive increase in the availability of image and textual data, for custom technologies, user data needs to be collected through structured user studies. This is a time–consuming process, which needs to rely on carefully–designed experimental protocols, and many users performing specific tasks with the sensors. Additionally, *noise sources in the real world* as reflected in the data need to be considered, which include different weather conditions, everyday use scenarios, proximity to other electronic devices. In such scenarios, the signal needs to be extracted from noise, but it can be difficult to model or even immediately

recognize all its potential sources. As part of the process, after the data is collected and before it is used for learning, signal pre–processing and feature construction should be performed.

Since the broader focus of this work is using such methods to enable user–centric devices, other considerations are also relevant. One aspect is the creation of interactive systems for real-world use, which means that the existing knitted sensor and the artificial intelligence component would need to interact seamlessly in real–time. For this reason, along the process, the necessary *deployment hardware* needs to be considered. Before creating applications for integration into our everyday clothes or environments, it is important to include *the end users' views* as well, and its ease of integration with existing hardware or in our everyday environments. Such studies can start by investigating more generally users' desired applications, expectations, and hesitations, to later continue to usability studies regarding each application or interaction. The section below discusses ways to address these challenges, constituting the contribution of this dissertation at the intersection of artificial intelligence and user experience.

1.4 Intellectual Contribution

This work builds the foundations for creating real–time, interactive systems that use minimalistic knitted sensors as an input medium. These systems are composed of lightweight and portable hardware, on which computational models can be deployed for high–accuracy recognition, serving as the bedrock for a multitude of future applications.

First, this work introduces algorithmic and deep learning–based recognition models for detecting location of touch and gestures on the sensors described here. The time–series data acquired through the few connections of the knitted sensor needs to be processed to map to a location of touch or a specific gesture, according to applications' needs. The classification algorithm should take advantage of the continuity of sampled frequencies over time, in order to be able to properly model the signal from few connections. Moreover, due to variability in the data, such as different users, conditions, finger placement and more, key presses of the same location should be represented in terms of essential patterns within them. To those ends, I utilize CNNs and LSTMs, which have found

application for many classification tasks: LSTM for sequence analysis, and CNN for capturing local features.

This process, in addition to algorithm and neural network architecture design, includes signal processing, feature extraction, data collection experiments, and software components for data processing. The use of machine learning aims to enable systems that adapt to the unobserved variations introduced by real–world environments. Subsequently, the real–world integration needs are investigated more in depth, with the purpose of understanding the conditions that could interfere with the adoption of this technology, including everyday use scenarios, such as sensor stretching, washing and drying. In addition, to investigate the potential of this technology from the end users' viewpoint, I also designed, ran, and analyzed a formative focus group study. More specifically, these contributions are detailed below:

- Designed Mixed–Source Description (MSD), an algorithm to detect invariances in the scale-space of multi–channel signals, and represent the signal in terms of its salient features and most important aspects.

- Designed Euclidean Levenshtein Distance (ELD), a generalizable distance metric which measures the similarity between two tensors of varying lengths, compatible with multi–dimensional time series data.

- Created a recognition model to identify precise location of touch on a single–yarn knitted sensor circuit, achieving 66% accuracy on a 36–button sensor without subject–specific calibration, while chance accuracy is 3%. This model relies on a statistical, frequency–based feature construction component and an LSTM neural network classifier.

- Designed and analyzed three user studies to move towards real–time, real–world knitted sensors. The studies enable the creation of a robust location sensing model; demonstrate its generalizability with new users; and show robustness under conditions of exposure to electromagnetic radiation and stretching of the sensor.

- Designed, ran, and analyzed a formative study with 32 participants structured as 8 focus

groups, to understand users' perceptions and the potential of this knitted sensor technology.

– Ran experiments to test the robustness of knitted sensors under everyday use conditions, such as stretching, and washing and drying.

– Created the foundations for an interactive gesture recognition system using a novel 4-connection knitted sensor design as an input medium, and a trained model for classification, deployed on an NVIDIA® Jetson Xavier™ NX system–on–module.

– Designed and analyzed three user studies using the 4-connection sensor for data collection. The data from these studies was used to create a CNN–LSTM gesture recognition model capable of classifying 12 complex gestures with 90% accuracy; demonstrate its generalizability with new users; and investigate its robustness with data collected while the sensor was being worn.

1.5 Document Organization

These considerations are jointly explored in several parts of this work, with each informing the next steps of the other. First, in chapter 2, background is provided on the sensor design process, the different design forms, and this technology is contextualized as part of ubiquitous computing. Chapter 3 discusses the algorithms and artificial intelligence components upon which the proposed computational models are based, as well as some other textile sensors that have applied machine learning to detect user touch events. The detailed description of this work's contributions starts by exploring the feasibility of building an accurate algorithmic computational model to represent the signals output from the sensor in chapter 4. Subsequently, chapter 5 introduces a touch location recognition model, a real–time signal generation component, and experiments that expose the sensor to real–world conditions. In chapter 6, the real–world integration aspects are explored more extensively, including a qualitative user study, more everyday use experiments, and preliminary experiments regarding the potential for gesture recognition. Chapter 7 finalizes the contribution of this work by introducing the basic building blocks of potential interactive systems: a new, four–connection sensor design form produced using the same technology as the other sensors discussed, but with

more information output from the system, and a deep learning model to recognize complex gestures, deployed on an NVIDIA® Jetson Xavier NX, a small, lightweight, and powerful embedded system–on–module. Chapter 8 concludes with a summary and discussion of future work. The explorations of this work range from enabling accurate interaction to usability considerations in the real world, and bring us closer to AI–powered, ubiquitous, real–time systems that enhance user experiences through touch–sensitive knitted fabrics.

Chapter 2: Context

Advancements in several areas of technology have enabled human–centered electronic devices to become smaller in size, less invasive, and more intuitive to use. Among others, examples of such sensors include brain–computer interfaces, activity tracking bands, augmented and virtual reality (AR/VR) devices, and textile sensors. They are designed to be harmoniously integrated into interactive environments and complement or inform human activity. The knitted capacitive sensors on which this work focuses follow similar principles. They are designed with reduced wiring and hard electronic components, and can be manufactured at scale, creating many possibilities for novel applications in textile sensors, and showing great potential toward becoming truly pervasive technology.

Section 2.1 provides background on ubiquitous computing and its impact, part of which are textile sensors, and particularly the knitted sensors which this work uses. These touch–sensitive fabrics are described in section 2.1.2, starting with an explanation of their design manufacturing techniques, and followed by illustrations of several existing prototypes. Subsequently, section 2.2 draws parallels between them and the stages of development of mobile computing, a cornerstone of ubiquitous technology. Next, some background on the signal processing method that is used with these types of knitted sensors is presented in section 2.3.

2.1 Ubiquitous Computing

Ubiquitous computing is the concept and associated field of study in computer science describing computers moving away from static desktop–centered settings and becoming immersed into everyday environments [135]. As a research discipline, it started with Mark Weiser in the mid–1980s, who criticized the concept of computers becoming users' objects of interest in themselves and proposed the opposite paradigm: that they should become "invisible" [140]. The embodiment of this design philosophy would free users from having to learn the interface of any new technology and allow them to look beyond it toward any new visions or goals. This means that computing would be seamlessly

integrated into our everyday lives, facilitating intuitive interactions.

When computers became small enough to allow portability, mobile computing emerged, with mobile devices being one of the first truly pervasive interactive devices [122, 57]. From an initial focus on minimizing the hardware and emphasizing the technical aspects, nowadays, the device design considerations have expanded to include usability, user experience, and interaction design. Cell phones, earphones, smart watches and other handheld and wearable computing technologies have become integrated into our lives to a considerable degree, with great potential for novel interaction designs remaining. Currently, the state of pervasive computing is such that we are surrounded by more computational devices than people [122], moving us closer toward Weiser's vision [140].

The history of mobile computing can be divided into several major periods of development [122]: *(1) portability*, focusing on reducing hardware size to ensure mobility; *(2) miniaturization*, to create new, significantly smaller form factors for personal use; *(3) connectivity*, for connecting multiple devices and users through wireless networks; *(4) convergence*, by integrating several functionalities such as mobile phones, music players, cameras and more into the same device; *(5) divergence*, consisting of designing mobile devices of special functionalities, which is the opposite practice of the previous phase; *(6) apps*, the period in which we currently are, focusing on creating content and designing a myriad of interactions with which users can engage, as well as intuitive ways to access the technology; *(7) digital ecosystems*, the emerging wave of considering the associated network of pervasive devices more holistically. As a result, the challenges of mobile interaction have evolved over time, starting with physical design, reducing device size while optimizing the screen display and keypads, and later moving to add–on features in the period of *convergence*, such as cameras, games, music players. Currently, the hardware remains mostly static, and the challenges have shifted to software applications and interaction design [122]. This development context is also relevant as a reference to consider that of related emerging technologies, such as wearable computing.

Wearable computing may be worn over or in clothing, directly on the human body, or may themselves be clothes [122]. Another important concept is that of *body–borne computing*, which includes wearables, portable devices like smartphones, and more generally any technology that is on or in

the body, for example implantable devices. An important difference between wearable computers or body–borne computers and portable computing, such as handheld and laptop computers, is that wearable computing by design contextualizes the computer so that the human and computer are intertwined, by placing the human in the feedback loop of the computational process. This process tends toward Mann's *Humanistic Intelligence* concept [92]. One of its main features is constancy of interaction, i.e., there is no need to turn the device on prior to engaging it, and another is the ability to multi–task. It is not necessary for a person to interrupt their activity to use a wearable computer, since it should always be running in the background, expanding, mediating the human's interactions, or passively waiting for input [122]. Examples include augmented, virtual, and mediated reality. Another prominent and emerging area of wearable computing with considerable potential for everyday life interactive, constant interaction is that of smart textiles, discussed below.

2.1.1 Touch–Sensitive Fabric

Textiles are among humankind's first inventions and have since evolved beyond uses for comfort and protection. Advances in materials, manufacturing, sensing, and computing have given way to novel interactive systems supported by textile touch sensors. The ubiquity of textiles, as well as their soft and flexible nature, makes them an appealing medium for tangible embedded interactions.

Fabric–based touch sensors have been investigated for over two decades in the field of human–computer interaction (HCI). In addition to fabric construction, they incorporate concepts from circuit design, signal processing, material properties, and recently machine learning. Textile touch sensors have been created through various manufacturing techniques, such as weaving [29, 55, 127, 7, 108, 143], embroidery [107, 42, 52, 53, 9, 94], and knitting [114, 141, 106, 30, 133] to investigate their potential as flexible, pervasive interfaces. The interaction potential of textile sensors has been demonstrated through the introduction of various application prototypes and use cases [101, 142, 53, 108]. However, for textile–based touch sensors to conform to users' expectations and have real–world viability, there are many practical considerations that must be addressed. One important aspect is designing toward scalable manufacturing to create textile devices that take advantage of mass–production techniques and that use commercially–available materials. Another is recognizing that

the physical demands regularly placed on textiles must not interfere with the device's performance. In particular, textile touch sensors must maintain integrity and sensitivity, while easily flexing, folding, stretching, and dynamically conforming to the human body in motion [40].

The wide variety of smart textiles that have been explored rely on different sensing and construction strategies. Many of them, especially woven smart textiles [108], use layers of knitted electrical traces sewn together as a grid, emulating hard electronics, which provides good touch localization ability. Other sensors have been produced using layers of fabric combined with thin layers of electronic components [114]. However, the multitude of wires or relatively complex, connected layers to sense input of some of these strategies may reduce durability and leave these sensors susceptible to damage from bending, especially at fabric–to–wire connection points. To address durability, connectorization, and production at scale, the textiles which I use in this work [133] rely on digital weft knitting, which enables the fabrication of a textile circuit with a single, electrically continuous trace. Carbon–suffused nylon yarn is used as the conductor, and it is combined with non–conductive yarns throughout the fabric structure. More details regarding their construction process and the various form factors possible are provided in the section below.

2.1.2 Minimalistic Knitted Sensors

This section describes the sensors on which this work focuses, starting with their construction process, to help elucidate some choices and challenges in creating knitted sensors and extracting information from them. Next, several design form factors producible through the same manufacturing process are described. The design philosophy of this technology tends toward hardware minimalism and flexibility of design. It is not oriented toward a particular application but aims to develop a platform for manufacturing and integrating touch–sensitive fabric into existing everyday environments. Given that fabric is so prevalent in our lives, applications based on this technology can rely on the knowledge users have in interacting with fabric, while experiencing added functionality.

Constructing Knitted Sensors with a Single Conductive Yarn

These fabric sensors are produced through digital knitting and rely on one conductive yarn and two external connections for sensing [133]. Digital weft knitting is a manufacturing solution that produces textiles using an array of continuous, intermeshed yarns. To produce touch–sensitive textiles, one of the yarns used in this process is conductive, which is knitted along a serpentine pattern. This simplifies the physical construction of the sensor, since the fabric circuit design is directly compatible with the weft knitting process and does not require any post–assembly. The carbon–coated yarn is combined in this process with regular, non–conductive yarn. To ensure reproducibility, and support production within a manufacturing ecosystem, commercially available materials are used to construct these textile sensors: carbon–suffused nylon [68] for the conductive yarn, and polyester for the non–conductive yarn. There are only two electrical connections per sensor–one at each end of the knitted electrical pathway–which simplifies the connection to sensing hardware and affords greater textile flexibility and connection durability. Different knitted design forms can be constructed using one continuous sensing yarn and two external connections.

Figure 2.1: Knitted touch sensor design workflow. Figure produced at CFF [3]: 1) Textile designers conceptualize sensor layouts while considering knitted circuit design guidelines. 2) The textile layouts are programmed using *Knit Paint* in the *Shima Seiki SDS–ONE APEX3* software suite, and compiled for the desired weft knitting machine. 3) The compiled programs are transferred to the configured knitting machine for assembly. 4) The completed sensors undergo testing and evaluation to assess performance. 5) The textile circuits are directly integrated into devices with no human intervention needed after assembly.

Figure 2.1 provides some insight into the textile production process. The design process for these sensors includes programming a specific design pattern, and then uploading it to the weft–knitting machine, which produces each sensor according to that pattern. The program *Knit Paint* is used in the *Shima Seiki SDS–ONE APEX3* knitting design software suite to program the loop structure for

each design. These sensors are produced using *Shima Seiki* SWG041N and SSG122SV computerized flatbed weft–knitting machines [6], which can produce small to medium–sized textiles, such as shirts or gloves. Digital weft knitting is a versatile, end–to–end textile manufacturing process that lends itself well to rapid prototyping. Textile designers can create finished pieces without the need for additional assembly using robust programmed instructions standardized across many digital knitting platforms. Weft knitting, unlike weaving, uses comparatively few yarns that do not separate at the fabric's edge and are continuous throughout the textile.

(a) (b) (c)

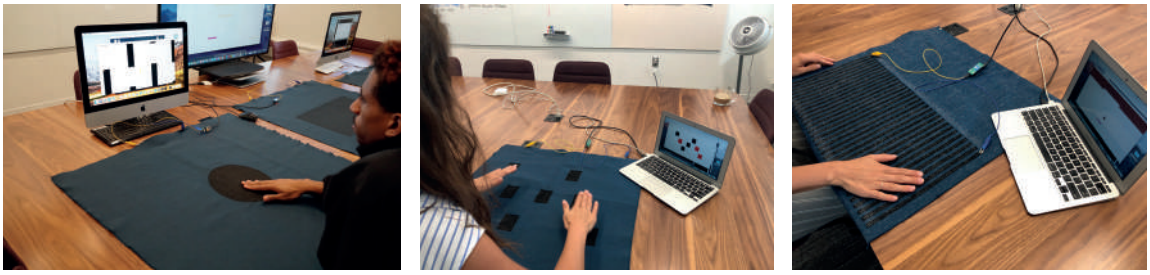
Figure 2.2: Examples of the knitted capacitive touch sensor created through digital weft knitting [3]: *(a)* A planar knitted fabric touchpad made from polyester and carbon–suffused nylon connected to small, lightweight hardware through only two electrodes at the yarn endpoints. *(b)* Small, interactive knitted components and sensing circuit integrated into a sleeve. *(d)* Knitted computer keyboard.

Design Form Factors

Through the same manufacturing process described above, a variety of knitted patterns can be produced, with some previously–constructed design geometries [133] presented in this section. The knitted samples descried below vary in patterns and dimensions, and each can be used as a *standalone knitted controller* or connected in a *system of knitted components* to create interactive devices. They include designs with finely–defined touch locations (fig. 2.2), three game controllers (fig. 2.3) which provide user input to different applications, and three media control sensors (fig. 2.4), which can be used individually or together to control an application. These components can also be produced in a variety of sizes, colors, and materials of non–conductive yarn. They illustrate examples of patterns that have uniform sensitive areas, and others with non–uniform ones.

Although the yarn routing of each pattern differs, each circuit maintains only two connections to external sensing hardware. The embedded hardware is interchangeable between knitted sensors. The

touch location decoupling method used in these proof–of–concept designs relies on the previously introduced method of *differential capacitive sensing* [133], described in section 2.3.2. These touch patterns are designed to explore the ability to recognize discrete position, amount of pressure applied, or continuous swipes based on their construction. The properties and design intentions behind each component are described below.



(a) (b) (c)

Figure 2.3: Design patterns and applications [3]: *(a):* The *Single–Button Game Controller* sensor controlling the *Flappy Bird* game application *(b):* The *Multi–Button Game Controller* prototype used for the *Whack–A–Mole* game; *(c)*: *Slider Game Controller* controlling the paddle position within the *Brickbreaker* game.

– *Knitted Touchpad:* Figure 2.2(a) demonstrates a knitted touch sensor, and is the main design which I investigate in terms of location identification. The button–like touch points are formed by exposing yarn on the fabric surface during knitting and are spaced approximately 1/2 inches apart. The touch points are knitted along a serpentine path. Two alligator clips connect the yarn endpoints to the sensing controller. This sensor geometry resembles typical keypads. The design can be scaled by increasing the length or width of the pattern within the *Knit Paint* program, to produce sensors of different sizes. Besides being used as a standalone sensor, this form factor can be incorporated into other fabric, for example sewn into a slip–cover, with wiring hidden inside the upholstery.

– *Sensing Sleeve:* Figure 2.2(b) illustrates a sensor knitted within a sleeve. The sleeve is created with the conductive yarn fully–integrated, without needing to sew the knitted components separately. A pocket on the inside of the sleeve can host any low–profile electronics necessary for interactivity. The sensing hardware attaches to the sleeve by fabric snaps and can easily

detach to allow the shirt to be laundered. The sleeve maintains flexibility and can fold and roll without damage to the fabric or wiring.

– *Knitted Keyboard:* Knitted sensors can be constructed with the conductive yarn structured in different patterns, according to the purpose of each application. The carbon fiber yarn is knitted to form an alphanumeric computer keyboard, illustrated in fig. 2.2(c). The touch points are knitted in the same serpentine structure as those in the touchpad, though the placement is staggered to mimic a standard keyboard. The buttons are again spaced approximately 1/2 inches apart to maintain the spacing of a full–sized computer keyboard. A textile keyboard can be made lighter, thinner, and more durable than a conventional keyboard. The keyboard could be integrated into a laptop sleeve or tablet cover.

– *Single–Button Controller:* This prototype, illustrated in fig. 2.3(a), has a large circular sensing area in the middle. It can be used as a controller for games that rely on pressing a button, as well as controlling pressure [133]. Pressure sensitivity can be a desirable quality for a game controller, that can possibly enable applications current hardware controllers might not. Its functionality can be demonstrated by using it as a controller for the *Flappy Bird* game, which prompts the user to guide a shape through gaps in a wall, with the shape rising or falling depending on the user's applied pressure.

– *Multi–Button Controller:* This design, as seen in fig. 2.3(b), is composed of several square–shape sensing areas. The primary interaction modality for this pattern is pressing one of the large sensing locations. This prototype was designed as a controller for a *Whack–a–Mole*–style game. Touching a button when prompted by the application (flashing red) triggers an in–game hit. Such sensors can be incorporated on different surfaces for a variety of functionalities and applications.

– *Slider Controller:* This pattern, illustrated in fig. 2.3(c) is intended to capture continuous input, such as *swiping motion*. This controller can be used for games that rely on a continuous sliding input, such as *Brickbreaker*, during which the user controls a horizontally–moving slate,

aiming to catch a bouncing ball.

– *Piano Keyboard:* This prototype, demonstrated in fig. 2.4(a), is a 25–button touch–pad knitted to emulate a 2–octave piano instrument. The design and spacing of its keys follow those of a regular piano keyboard. It is intended to be a familiar and intuitive interface, with the added flexibility and light weight allowed by the fabric material. This prototype is an example of a knitted sensor with irregular size and spacing of buttons. The *piano keyboard* is used as a MIDI controller for the *GarageBand* application.

– *Volume Slider:* A 22–row slider, illustrated in fig. 2.4(b), was knitted to control the volume in *GarageBand*. It relies on a swiping gesture for continuous input control, similarly to the *Slider Game Controller* described above, but its proportions are smaller. Both its form and the sliding mode of engagement aim to be intuitive and conform to users' expectations of a volume controller. Rows are knitted approximately 1/8 inch apart, to maintain contact with the user's finger as it slides along the pad.

– *Media Control Buttons:* The button pad depicted in fig. 2.4(c) contains three buttons serving as discrete inputs to the same application, while graphically showing their functionalities in familiar forms: rewind, pause/play, and fast–forward.

(a) (b) (c) (d)

Figure 2.4: Intuitive design for knitted components that work together to control the same application [3]. *(a)*: The Piano Keyboard. *(b)*: The Volume Slider. *(c)*: The Media Control Buttons. *(d)*: The piano keyboard, volume slider, and media control button touchpad connected in a system. The sensing hardware is networked via I2C, with a single sensing controller directing communication among all three devices. The master I2C device connects to a MacBook Air via USB MIDI to interface with *GarageBand*.

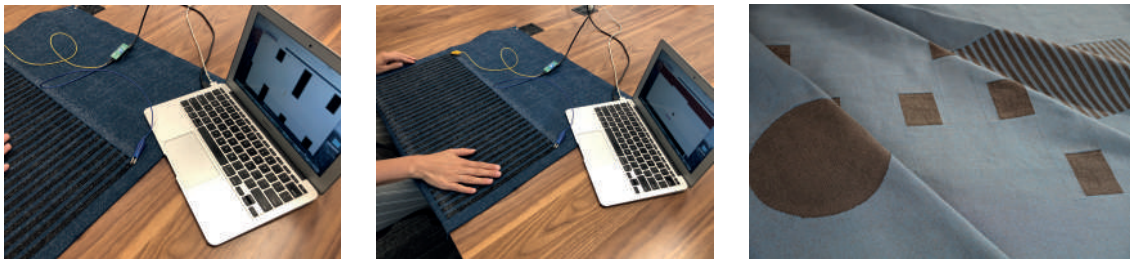
After the description of the fabric component manufacturing, and the illustration of the process versatility through several form factors, the next section contextualizes the development cycle of

minimalistic knitted sensors as part of the larger ubiquitous computing ecosystem.

2.2 Minimalistic Knitted Sensors as Ubiquitous Technology

The knitted sensors described in section 2.1.2 have the potential to embody the principles of ubiquitous computing, briefly discussed in section 2.1, envisioning computing harmoniously integrated into our everyday life, inviting effortless and intuitive human interaction. This technology was designed with those underlying principles at its core, and its developmental stages can be paralleled with those of mobile computing, even though some concepts are expressed in different forms.

Many of the phases of mobile computing happened linearly in time, while knitted sensors and smart textiles in general, attempted to incorporate the principles of ubiquitous computing earlier in their design stage, most probably since the pervasive computing field was more mature at that point, compared to the emergence of mobile computing, which is considered as one of the first truly pervasive technologies [122]. Additionally, today, some of the developments are happening in a more circular fashion, based on rapid prototyping. Here, I revisit the main stages of the emergence of mobile computing, each of which was based on a fundamental quality of ubiquitous computing, but they are discussed in the light of the knitted sensors technology upon which this dissertation is built.



(a) (b) (c)

Figure 2.5: Exploring the parallelism to mobile computing regarding *convergence* (*a* and *b*) and *divergence* (*c*) in knitted sensors [3]: *(a):* The *Slider Controller* used to control the *Flappy Bird* game. *(b):* The same knitted sensor as in *a*, used to control the *Brickbreaker* game. *(c):* Knitted fabric prototypes of varying patterns easily producible through the same process for specific functionalities.

– *Portability:* Fabric has the inherent quality of portability, and knitted sensors can be designed to have a small form factor, if desired. The few connections to external hardware and the fact that the sensor design does not rely on several layers of fabric and electrodes also help

portability. Additionally, the size of the attached processing hardware for the existing circuits is already small, as can be seen from fig. 2.2(a). It is necessary to ensure any new components added still comply with this requirement.

– *Miniaturization:* In mobile computing, this principle was expressed by introducing smaller form factors for personal use. In this case, in addition to the previously discussed aspects that make the knitted sensor technology portable, the patterns of interaction can become minimal as well. Some of the design forms factors, such as the touchpad on which I focus (fig. 2.2(a)), are fine and compact. Moreover, a small number of its interactive components can be integrated into clothing, such as the sleeve illustrated in fig. 2.2(b).

– *Connectivity:* This stage in mobile computing focused on connecting multiple devices and users through wireless networks. That technology is already in place, and knitted sensors can be easily integrated into existing networks. They can be envisioned to connect to mobile phones, laptops, and other electronic devices, in addition to other knitted sensors connected in a system, depicted in fig. 2.4(d).

– *Convergence:* At this point in the mobile computing development, several functionalities were integrated into the same device. There is no direct parallel between this functionality and textile sensors, but the principle can be approximately translated to designing the same rather generic sensor with interactive patterns and enabling different applications by mapping different meanings to the same physical design (fig. 2.5(a) and fig. 2.5(b)).

– *Divergence:* This principle is essentially the opposite of *convergence*, since in mobile computing, it involved producing devices with specific functionalities. Rapid prototyping in manufacturing these sensors enables the creation of different design patterns, which can be specialized in their features, as seen in fig. 2.4(d) and fig. 2.5(c).

– *Apps:* Mobile computing has been seen as currently focused on the stage of content creation, primarily through mobile applications [122]. Analogously, this work aims to enable the creation of various applications using minimalistic knitted sensors with which users can engage

intuitively, with some prototypes illustrated in fig. 2.5(a) and fig. 2.5(b). In order for this concept to become a reality, however there are several challenges to be overcome, primarily related to the computational models necessary to interpret the signal generated from the contact between the user and the sensors' interactive components.

– *Digital ecosystems:* This emerging stage in the development of the filed of ubiquitous computing aims to consider the associated network of pervasive devices more holistically. In addition to future considerations of knitted sensors as part of such devices, this smart textile technology is itself a platform for producing related sensors (fig. 2.5(c)), part of the larger area of textile touch sensors. Therefore, considerations regarding constructions within this smaller ecosystem are useful, and have been considered from a manufacturing standpoint. Computationally, it is necessary to examine that aspect, but it should more fundamentally be the focus of future work, especially once the current stage of enabling content is finalized.

Similarly to the challenges of mobile interaction, which started with physical designs, continuing with reducing device sizes, and currently having evolved to emphasize software applications, the next phase in developing ubiquitous, minimalistic, knitted sensors is application and interaction design. However, in order to arrive to content creation, first, the underlying computational technology of this smart textile manufacturing platform needs to be constructed and solidified, with capabilities such as touch location and gesture recognition. In addition, user perceptions and desires regarding such potential applications need to be explored, as well as the sensor performance in conditions closer to the real world.

Section 2.1.2 presented some existing prototypes, which rely on the circuit design and signal processing methods that will be described in the next section. Even though high–accuracy touch location identification and gesture recognition is not possible using such techniques, they offer approximate touch localization.

2.3 Circuit Design and Signal Processing

The sensor construction described section 2.1.2 is incompatible with the circuit layout used for conventional sensing matrices, which require isolated traces. Therefore, in order to sense distributed touch across a knitted fabric, input needs to be measured across a continuous yarn integrated within the textile. Electronics are connected at the two yarn endpoints to complete the circuit. The fabric circuit, consideration related to knitting, and the sensing method, each described in more detail in the following subsections, were developed by Vallett [134] to be specifically compatible with the above–mentioned manufacturing method.

2.3.1 Sensor Knitting Considerations

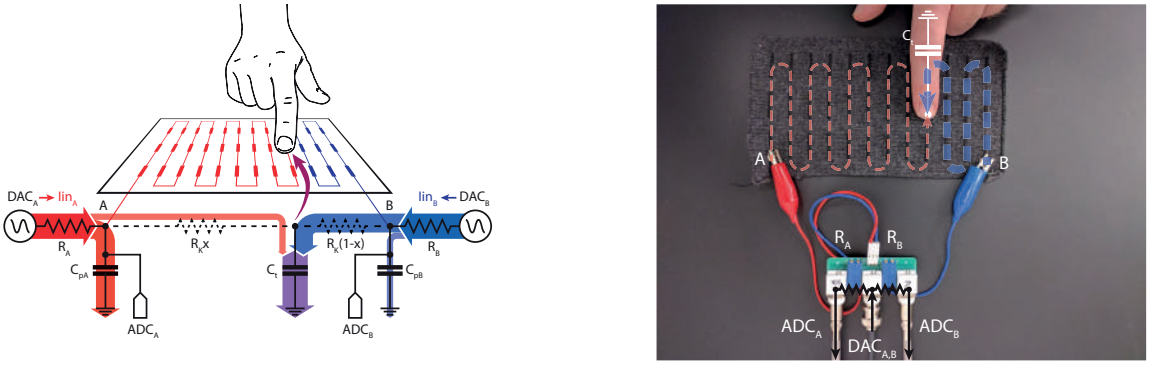
Touch location on the sensor is inferred based on the resistance of the conductive pathway. Controlling its resistance is crucial to ensuring separable touch data, tuning external current–limiting resistors, and predicting signal behavior. The capacitance induced by human touch varies within the range of pico–Farads. Accurately measuring small values of capacitance and fine changes in current across the conductive pathway requires a high yarn resistance. The overall resistance of the pathway is set through modifying the yarn density (denier) or through modifying properties of the knitted pattern. Yarn denier, which is the mass or density of a given length of spun yarn, can be altered to produce an appropriate cumulative resistance. In the case of the carbon fiber trace, the yarn may be represented as a wire of homogeneous composition having a linear resistance, R. The wire resistance is a function of the resistivity, ρ, an intrinsic material property, its length, l, and fiber cross–sectional area, A. The resistivity is set during manufacturing and varies by production lot. The yarn resistance is proportional to its length, and inversely proportional to its cross–sectional area. In practice, significantly decreasing the denier proves difficult to knit due to the fragility of the single–strand monofilament yarn. Combining a lower denier monofilament with a non–conductive yarn improves knitting but sacrifices homogeneity and complicates yarn interconnections.

Specifying the overall sensor resistance through loop formation is not as straightforward as specifying the fiber resistance through length or thickness. The loops formed during the knitting process

create a complex, interconnected mesh of resistors [150]. Understanding the interactions between the loops is crucial to better predicting the touch sensor resistance during the design process. Specifying the width and height of the touch points alters the resistance intuitively. Widening the trace decreases the resistance by increasing its cross–sectional area, while lengthening the trace increases the resistance. The fabric circuit and its sensing method, discussed below, needed to account for these manufacturing–related considerations.

2.3.2 Differential Capacitive Sensing (DSC)

As described in section 2.1.2 and section 2.3.1, the digital weft knitting process can produce complex textiles by integrating conductive and non–conductive yarns. All touch locations along the surface of the fabric can be mapped to a linear arrangement of points on the sensing yarn, which can be part of an electrical circuit connected with other hardware. At the two endpoints of the carbon–coated yarn, two electrodes can be connected. *Differential capacitive sensing (DCS)* [133, 134] has been previously introduced to indicate an approximate position of touch, and is summarized below.



(a) (b)

Figure 2.6: *(a)* Circuit diagram of the fabric capacitive touch sensor and sensing circuit: The knitted resistor, R_K forms two series–connected resistors at the contact point. *(b)* An overlaid image showing a real knitted touchpad and connections to sensing hardware [134, 3]

Figure 2.6(a) depicts current flow from either endpoint to a touch location on the fabric. A synchronized voltage waveform is input at either end of the fabric using a *Digital–to–Analog Converter (DAC)* at points DAC_A and DAC_B and passed through two equivalent current limiting resistors, R_A and R_B. The current output at points A and B branch both into the knitted circuit and towards the voltage measurement circuit. The parasitic capacitance of the measurement circuit contributes

to the response recorded by two *Analog–to–Digital Converters (ADCs)* at points ADC_A and ADC_B.

Figure 2.7 shows example input and output voltage waveforms recorded by an oscilloscope. Figure 2.7(a) depicts the resting state of the voltage signals when no touch is present. Figure 2.7(b) shows the attenuation caused by touch occurring near the left (A) electrode. The capacitance induced by touch decreases the peak–to–peak amplitudes of both outputs and produces a greater phase shift. The electrode closer to the point of touch exhibits a more pronounced attenuation and phase shift. If the touch event occurred closer to the right (B) electrode, we would expect a response analogous to that in fig. 2.7(b), but with the right electrode having greater attenuation. In the case of contact occurring in the center of the circuit, the response would look similar to that shown in fig. 2.7(a), since the attenuation from both electrodes would be approximately equivalent. Figure 2.7(c) shows the effects of electro–magnetic interference (EMI) in addition to contact. The difference in peak–to–peak amplitude is present to signify touch but obscured by high–frequency distortion. EMI is a common source of noise in electrical circuits.

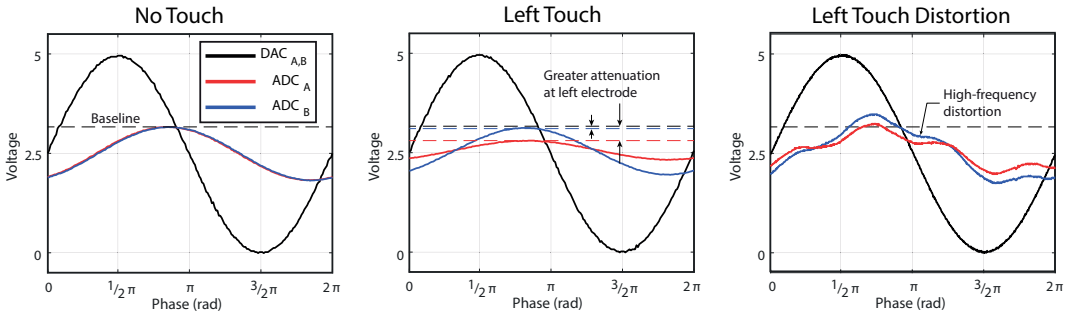


Figure 2.7: Example input and output voltage waveforms: *(a)* Input and output without touch. *(b)* Touch occurring near the left (A) electrode. *(c)* Left touch distorted by a nearby EMI source [134, 3].

Conventional capacitive sensors that measure attenuation or phase shift do so by directly tracking the voltage envelope or measuring the time difference between crossing voltage thresholds. Measuring phase offset with minimal distortion yields measurements that are relatively consistent. However, in the presence of additive interference, the variance of phase increases considerably.

This signal processing method offers some information regarding the location of touch on the sensor, however its accuracy is limited to identifying approximate regions where touch occurred

rather than fine locations, such as the buttons on the touchpad. My work focuses on identifying touch location more accurately, as well as investigating conditions to bring these sensors to the real world. Additionally, it extends to recognizing gestures on similarly designed sensors. It is worth noting that the signal as captured through *differential capacitive sensing* is prone to distortion from different sources, a few of which are briefly addressed in the next subsection, and Bode analysis, described subsequently helps address some of them.

Addressing Distortion in Single–wire Touch Localization

Touch sensors that discretize input across continuous conductors by detecting changes in electric field are susceptible to additive interference [151], which can reduce localization accuracy. The knitted fabric circuit described above uses a large end–to–end resistance to source and then measure fine changes in the current differential recorded at each endpoint of the conductive yarn pathway. The resulting low current is vulnerable to small fluctuations in voltage introduced by nearby external power sources or from contact by the user. These fluctuations distort measurements of the transient waveforms. Additive interference originates from both device–centric sources like *power supply ripple* and external sources like *electromagnetic interference*. In measuring touch at a single location using capacitive sensing, additive interference decreases signal–to–noise ratio and distorts touch input identification. In this way, otherwise insignificant sources of uncertainty, produce considerable measurement variance that inhibits practical use.

Bode Analysis

Bode analysis [18] used with differential capacitive sensing can be a potential solution for removing recorded interference. It is an extension of Fourier analysis, which transforms a measured time–domain signal into frequency–domain components. Bode analysis compares both the ratio of a system's Fourier transform complex magnitude output to its input, and the phase difference between output and input. The frequencies present in an input will be present in an output. Given an input with a known frequency composition, it is possible to reconstruct the output while bypassing additive frequencies.

This signal processing strategy described in section 2.3.2 is an important first attempt to char-

acterize human touch input events, however it offers only approximate touch location identification. To arrive to content creation, first, the underlying computational technology of this smart textile manufacturing platform needs to be constructed and solidified. The subsequent chapter on related work focuses on describing the context of the algorithmic and artificial intelligence components of the computational models introduced in this work, and some real–world considerations of smart textile implementations.

Chapter 3: Related Work

This chapter provides some related work to contextualize the innovations of different aspects of this work, since its main contribution is unlocking the power of minimalistically designed knitted sensors to enable interactive applications in the real world. Artificial intelligence has had a considerable impact on the world around us, transforming many areas of society, including healthcare, finance, robotics. It has the power to also transform the future of the knitted sensors on which this work focuses. Their design strategy includes reduced hardware and external wiring in favor of usability and ease of integration; however, this means that they need more refined computational models to interpret human input. Algorithmic and artificial intelligence approaches are necessary to model the variability of the external world. Moreover, for applications with touch–sensitive knitted fabric to become realistic, users' perspectives should also be considered, as well as these sensors' exposure to everyday life conditions. This work is positioned at the intersection of algorithms, machine learning, quantitative and qualitative user studies, and system design considerations.

Section 3.1 describes some relevant algorithms upon which the computational aspects of this work are based, starting with algorithms representing a signal in terms of its scale–space, and continuing with several distance metrics used with times–series data. Subsequently, section 3.2, provides a brief overview of the most relevant developments in artificial intelligence, primarily focusing on the neural network architectures that are used for the recognition models which this work contributes, followed by applications of machine learning algorithms in different fabric–based sensors. Additionally, some studies related to other fabric sensors in as they relate to everyday life conditions and user preferences are also discussed.

3.1 Relevant Algorithms

This section describes some existing algorithms, based on which this work introduces new ones for compatibility with multi–signal sensing, detailed in chapter 4. These algorithms are then applied to

signals generated from human touch on knitted capacitive sensors. Scale–Invariant Feature Transform (SIFT), described in section 3.1.1, serves to represent images in terms of their most salient and invariant features. Section 3.1.2, discusses several distance measures, which are used to compare sequences, especially focusing on the ones designed for time–series data.

3.1.1 Scale–Invariant Feature Transform (SIFT) Algorithm

Scale–invariant feature transform [89, 90] is a feature extraction algorithm, originally emerging from the computer vision community, which aims to detect and describe local features in images. Underlying its working is the concept of scale–space. Scale–space is a framework for multi–scale signal representation, which handles structures in signals, such as images for example, at different scales. The scale–space representation is a family of objects generated by varying one parameter of the original: the size of the smoothing kernel, which serves to suppress fine details of the object in each smoothed representation [74, 87, 111]. The scale parameter t, indicates that image structures of spatial size smaller than about $\sqrt{t}$ have been blurred. This process is analogous to viewing an image at different distances: further away, the most noticeable aspects are the general shapes of bigger components, while the closer one gets, the more noticeable smaller details become. Gaussian scale–space is the most widely used type of scale–space. It allows for scale invariance in visual operations to account for size differences in image data, since real–world objects could have a variety of sizes, and may be captured through a camera from different distances [86, 88].

Figure 3.1: This figure, obtained from the original SIFT paper [89], illustrates how applying this algorithm could enable object detection from a complex scene by focusing on the most salient aspects of signals, the images on the left.

In order to perform object recognition on an image, the most salient points of the object can be extracted to characterize it. This is what SIFT was designed to do. This object description can then be used to find a match of that object in images containing other objects as well, some including clutter and partial occlusion, since the SIFT descriptors are invariant to uniform scaling, orientation, illumination changes, and partially invariant to affine distortion [89, 91]. These qualities are essential to the algorithm's robustness with real–world data. SIFT starts with generating an image's scale–space representation, continuing to extract salient aspects of the image relying on its the Gaussian–blurred representation. The second stage is computing Difference of Gaussian (DoG) on the smoothed images, and then extracting key locations, which are the points with minimum or maximum values in their surroundings. Subsequently, orientations are assigned to each key location, which enables rotation invariance. Then, each key location is described in terms of its surrounding points, to produce features that are distinctive, descriptive, and to some extent, invariant to illumination, viewpoint, and other aspects. An example of the usefulness of this algorithm in an object detection task is illustrated in fig. 3.1.

SIFT has found many applications, such as in robot visualization and mapping [118, 34, 35], 3D modeling, recognition, and tracking [37, 46], human activity recognition [37, 117, 77], and human brain Magnetic Resonance Images analysis [129]. I adapt SIFT to time–series signal data output from knitted sensors due to human activity by creating the *Mixed–Source Descriptors (MSD)* algorithm, detailed in section 4.4.

3.1.2 Distance Measure Algorithms

Many applications require determining the similarity between sequences, and similarity measures have been used, among other areas, in early speech recognition [95, 113], object tracking [24], financial data analysis [8, 36], genetic sequencing [83], character matching and recognition [32, 128], and unsupervised learning on time–series data [98, 51, 15, 70, 33]. Some of these measures rely on symbolic representation of sequences, such as the Levenshtein distance [82], and its variations. For time series data similarity, pioneering work from Agrawal et al. [8] used Euclidean distance. The Euclidean distance is a metric, which is a desirable quality, especially for indexing databases. Eu-

clidean distance, however, is sensitive to noise and does not handle local time shifting. In addition, the research on the similarity of one–dimensional time–series signals [8, 73, 147] does not directly translate to multi–dimensional sensor signals produced from human activity. These signals may have varying lengths, outliers due to noise and disturbances, or similar patterns may appear in different parts of them [24].

Several distance measures have been proposed that address some of these issues. For example, Dynamic Time Warping (DTW) [17] represents a time series signal as a "stretched" or "compressed" version of another; it does not require the sequences to be of the same length; and it handles local time shifting. Longest Common Subsequence (LCSS) [138], and Edit Distance on Real Sequence (EDR) [24] also address time shifting and different sequence lengths, however, similarly to DTW, they are not distance metrics, since they violate the *triangle inequality* condition. EDR is designed to be more resistive to outliers, since it quantizes differences between sequence elements to 0 or 1. Edit Distance with Real Penalty (ERP) [23] is a distance metric in addition to handling local time shifting and different sequence lengths, but it might be sensitive to noise relative to measures like EDR [24]. Both EDR and ERP are time–series adaptations of the original string edit distance, which is described below. Subsequently, a brief overview of DTW is provided, due to its popularity.

Levenshtein Distance

The Levenshtein Distance [82], frequently called the *edit distance*, is a general dynamic programming sequence distance metric [23], typically used for string comparison. In that scenario, it refers to the number of edits necessary to convert one string into another, where an insertion, deletion, or conversion to another character has the same cost of 1 edit. Two strings a_n and b_m are compared character by character in a matrix. If two characters a_i and b_j are the same, the cost of that action is 0; if an edit is necessary, the cost of that particular change becomes 1. The cost of converting the string characters up to and including a_i to b_j is calculated by taking the minimum cost of the conversion before reaching one character before the current, and adding the cost of the last step, if any. An example is illustrated in section 3.1.2.

This distance metric has been used in several HCI applications [109, 27], in optical character

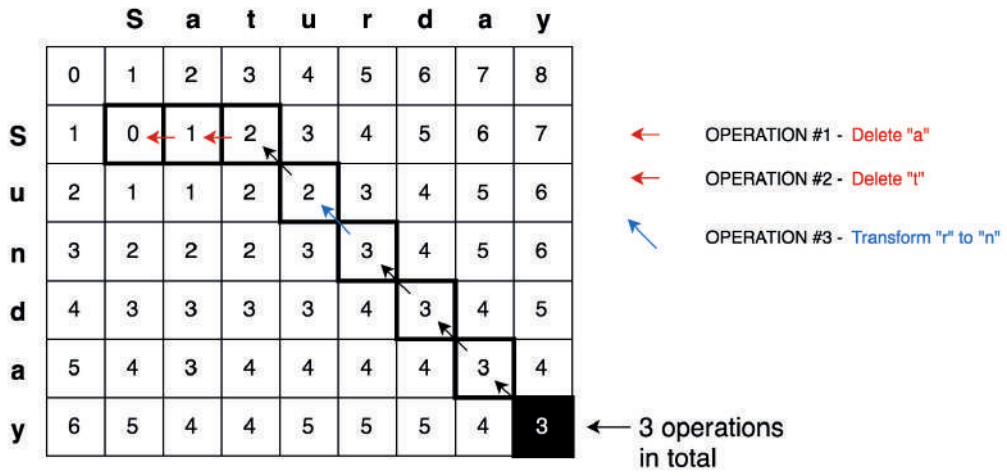


Figure 3.2: An example of the Levenshtein distance algorithm used for string comparison. Two strings are compared character by character to calculate the minimum number of edits necessary to transform one into the other. This image was obtained from [130].

recognition and string correction [58, 116], and linguistic analysis [67]. However, it returns only values of 1 or 0, and it does not compare arbitrary vectors or tensors, only the number of edits as the difference between two strings. In the case of time–series tensors, they would have to be identical for the Levenshtein Distance to be 0, which is incongruent with the nature of signals from sensors or their representations. Alternatively, one could define a threshold over their difference to output a 0 or a 1. However, I define the *Euclidean Levenshtein Distance* in section 4.5, which aims to preserve additional information related to the difference between tensors.

Dynamic Time Warping

Dynamic Time Warping (DWT) [17] is another well–known algorithm which measures the similarity between two sequences, potentially unequal in length. DTW has effectively been used in time–series analysis since it is capable of capturing similarities even when the two compared sequences are shifted or distorted in time [120]. A warping path W maps the elements of the two sequences to each-other to minimize the distance between them. It calculates this optimal match, under the following restrictions: 1) *boundary condition:* the first and last elements in each sequence need to map to each other respectively; 2) *monotonicity condition:* the ordering of elements in each sequence is preserved, with the mapping of the indices from either sequence to indices from the other needing

to be monotonically increasing; 3) *continuity condition:* the path transitions must be limited to adjacent points in time.

This technique is particularly useful when one of the sequences is a non–linearly transformed version of the other in time. DTW is a distance–like quality, and not a distance metric, since it does not guarantee the property of *triangle inequality* to hold. In DTW, each point in one sequence matches one or several corresponding points in the other, making the algorithm more resilient to varying sampling rates.

The algorithms described here are the foundations for important components of the computational models subsequently built in this work, designed to analyze and represent the signals generated from the textile sensors. However, these algorithms are also deterministic, while the behaviour of these sensors as they interact with various users and real–world environments is a stochastic process. Therefore, these algorithms alone would not be capable of capturing all the information necessary to model the sensors' functioning in everyday life, where they would need to be integrated for this technology to be truly ubiquitous. For this reason, it is necessary to introduce stochastic elements through machine learning (ML), a sub–field of artificial intelligence (AI), both of which are discussed in the section below.

3.2 Artificial Intelligence

Artificial intelligence is flourishing, both as a fundamental research field and with its many applications in areas such as medicine, robotics, and science [45]. In its beginnings, it emerged as a discipline capable of solving problems that are typically difficult for humans, but relatively easy for computers, problems that can be described formally and mathematically. However, it seems that its true challenge lies in performing tasks that are automatic or intuitive for most humans, but relatively difficult to formalize, such as vision or language recognition. Such functionalities in humans rely on the exposure to and the synthesis of large amounts of knowledge about the real world, which cannot be solved through hard–coded statements and formal definitions alone [45].

Machine learning emerged to bridge that gap by enabling computers to learn from experience through data from the real world. Machine learning is the sub–field of artificial intelligence focusing

on building algorithms, which rely on real–world examples of the phenomenon they aim to describe, to produce useful results. Such examples can directly come from the outside world, or the raw data can be subsequently processed by humans or other algorithms [20]. Ultimately, a successful learning algorithm would be able to generalize to samples not introduced in the original example dataset, making decisions that might appear subjective [45]. However, the performance of machine learning algorithms will be affected by the data representation. An example regarding signal data is representing it in time domain versus frequency domain, with different aspects of the signal becoming prominent in each representation. Many problems can be solved by using machine learning algorithms coupled with domain knowledge, which can translate into the right data representation. However, there are other problems where the relationships among different aspects of the data are too complex or unknown, thus making it difficult or even impossible to identify the features for an effective and comprehensive representation.

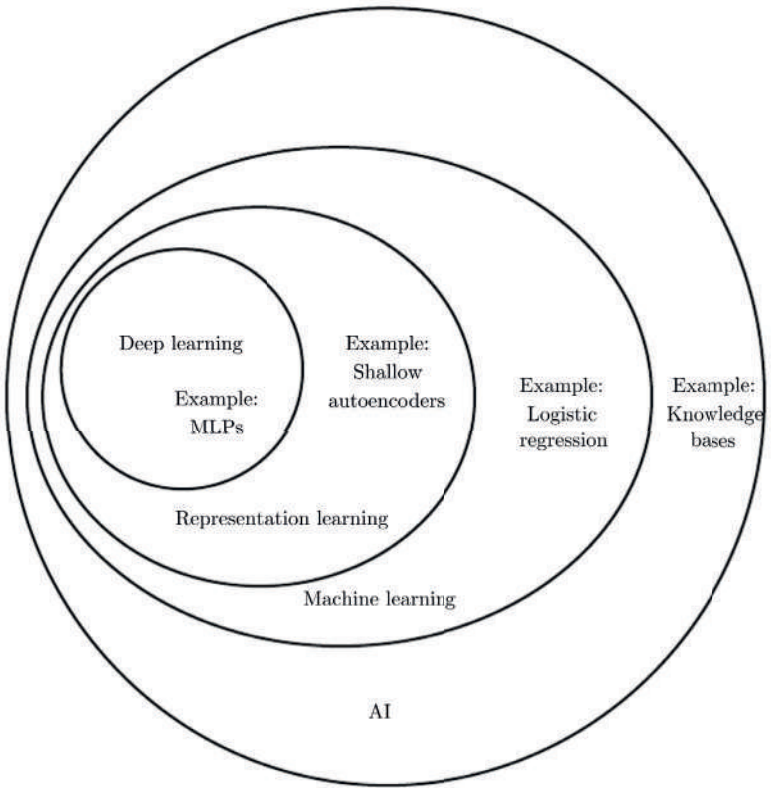


Figure 3.3: This diagram shows how the concepts described here are nested within each–other. Artificial intelligence is the broader field, and deep learning, its narrowest sub–field. This image was obtained from [45].

Representation learning, which is the strategy of using machine learning to learn the data representation itself, rather than only the mapping from the data to the output, could be used to tackle this issue. This solution allows artificial intelligence systems to adapt to new tasks with little human intervention, and typically leads to better results than hand–crafted features. When learning features, one goal is also to distinguish among and extract the different factors of variation, or aspects that influence the data as it is captured in the real world. Such factors may not be directly observable in the real world, but many times can be forces, concepts or abstractions that help disambiguate the complex data variability. Examples in images are time of the day which can affect direction of sunlight, camera lens, position of the observed object and many more. Several applications need these factors of variation disentangled, many times to discard noise or those irrelevant to the task. Nevertheless, it might also be difficult to extract such abstract concepts from the data, that would require human–level understanding of it, thus leading back to the problem representation learning aimed to solve [45].

Deep Learning emerges as a solution to this essential representation learning problem by creating a hierarchical structure of conceptual representations where more complex concepts are expressed in terms of simpler ones, with levels of abstraction increasing with complexity. The relationship between AI and its sub–fields is illustrated in fig. 3.3. An example of this representation is detecting an object in an image by first capturing lines, then corners, and subsequently more complicated shapes, shown in fig. 3.4. The simplest and most famous example of a deep learning is the *multilayer perceptron (MLP)* network, which can be considered a function produced by the composition of simpler functions, and after the application of each function in this sequence, a new representation of the initial data is produced. Through representing the complexity of the real world as nested hierarchies of concepts, deep learning offers great potential into integrating computers into everyday environments through a multitude of applications, enhancing the abilities of computing systems to adapt to our needs. Over the years, deep learning has acquired and adapted information from several fields, such as neuroscience, statistics, and applied mathematics. Recently, it has shown a stupendous growth in popularity, interest, and investment, since advancements toward enabling

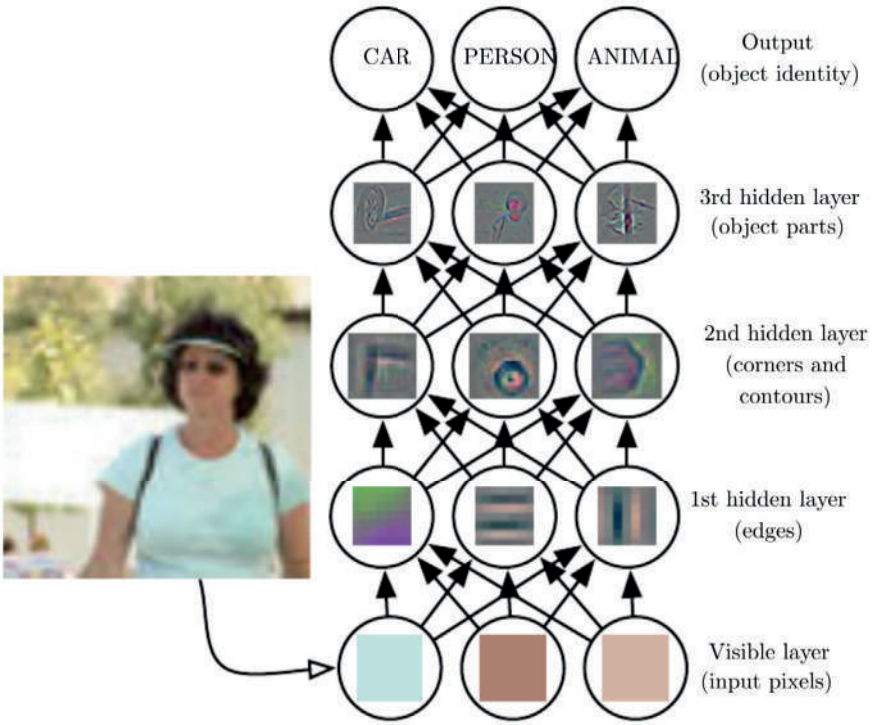


Figure 3.4: This image, reproduced from [149, 45], illustrates how more abstract concepts are built using as a foundation simpler ones in a deep learning model. The visible layer represents the raw input data, while hidden layers are the inner layers of the network which build the complexity and abstraction of the model, until the output layer is reached, which maps the data to a category.

powerful computers, the increased data availability, and the emergence of better algorithms to train deep networks fast, have made it more viable to be implemented in real–world applications [45]. Below, I describe two types of deep learning architectures that have had a great impact into solving real–world problems and which can be used to capture and meaningfully represent time–series signal data.

3.2.1 Convolutional Neural Networks

A convolutional neural network (CNN) [38, 79, 80], as described by GoodFellow et al. [45] is a specific kind of neural network, especially useful in processing data that has a well–structured topology, similar to a grid. Examples include images, which can be considered 2D grids of pixels, or time series data, structured as a 1D grid of regularly sampled instances. CNNs use convolution instead of general

matrix multiplication in at least one of their layers, an operation which is used to automatically extract features from the data. By using CNNs, there is a considerable reduction in the number of parameters of a deep neural network without much loss in the quality of the model [20].

When the input layer is convolved with the kernel, the feature map is produced. This is a convenient aspect of convolutional neural networks, because of the reduced reliance on domain knowledge. Together with the relatively low processing time and space requirement, and high accuracies achieved, this property has contributed to the popularity of CNNs. In CNNs, convolution is discrete, and can be viewed as a multiplication of the input image by a matrix. This matrix is constrained to have some entities equal to others, and is usually very sparse, because the kernel is typically much smaller than the input image. Convolution of the image with the kernel produces small, meaningful features, therefore fewer parameters need to be stored. This decreases memory requirements, and improves the model's efficiency. In CNNs, frequently, several of the parameters used in the model are shared among more than one function, unlike in traditional neural networks where each weight is used exactly once during the computation of the output layer. Therefore, CNNs learn the same set of parameters in different locations, which further decreases storage needs of the model. In time series, convolution is usually applied by shifting the kernel on the input data points by one measurement at a time, which enables the creation of a timeline showing where the feature captured by the kernel happened in the input.

The typical convolutional network, demonstrated in fig. 3.6, works by first performing convolutions in parallel to create a set of feature maps (fig. 3.5), then by running each of those maps through a nonlinear activation function, and lastly by running each output of the previous layer through a pooling function. A pooling function determines the output of each individual location on the feature map to be a number that summarizes nearby outputs. Typical pooling functions include reporting the maximum, or the average of the set of outputs. One of the main effects of pooling is invariance to small translations. This can be a desirable property of a model if we are more concerned with the existence of a feature rather than its exact location, and we care about improving the computational efficiency of the network. However, some tasks might require the precision that pooling reduces, in

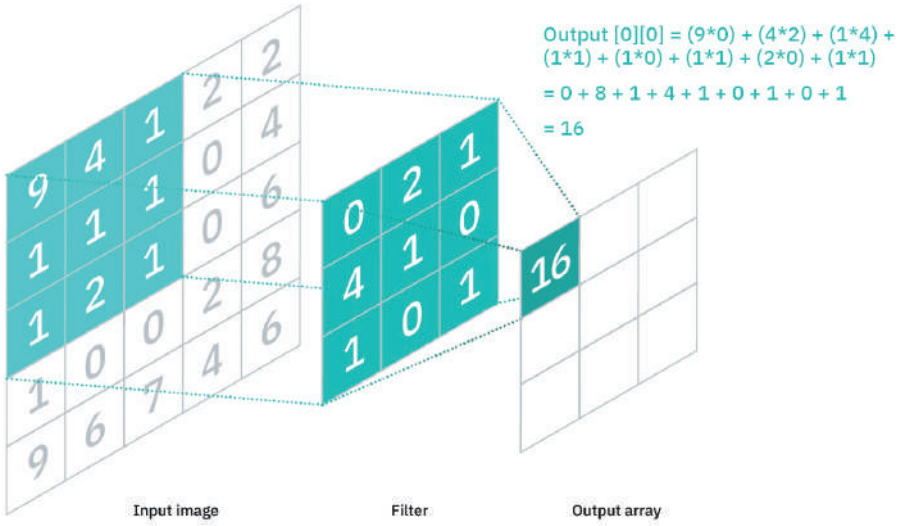


Figure 3.5: This image [66] illustrates the convolution operation on a two–dimensional image. The filter is convolved with the image using a sliding window strategy, producing the values in the output array.

which case using a pooling function would increase the error rate. Another important functionality of pooling is enabling the network to handle multiple sizes of inputs, since the pooling function can adapt and no matter what the size of the data it receives is, it can give the output layer a fix sized object [45]. After pooling, a fully–connected layer maps to the output.

Convolutional networks are the epitome of artificial intelligence inspired by the brain, since their design principles were drawn from the visual system. This process started with the investigations in the mammalian brain of Hubel and Wiesel [62, 60, 61], and the CNN design focuses on the *primary visual cortex*, capturing three main properties of it: the spatial arrangement with the two–dimensional structure; the *simple cells*, whose behaviour can be approximated by linear functions; and the *complex cells* which, in addition to the functionalities of the *simple cells*, are also invariant to feature positional shifts, a functionality which pooling aims to capture. These visual system guiding principles, nevertheless, are very far from the biological system itself. Moreover, neuroscience offers no guidance on how training should happen. That and other aspects are addressed by integrating mathematics and algorithms into the construction of CNN models. Many versions of such systems have been proposed and implemented, each focusing on specific applications or aiming to improve

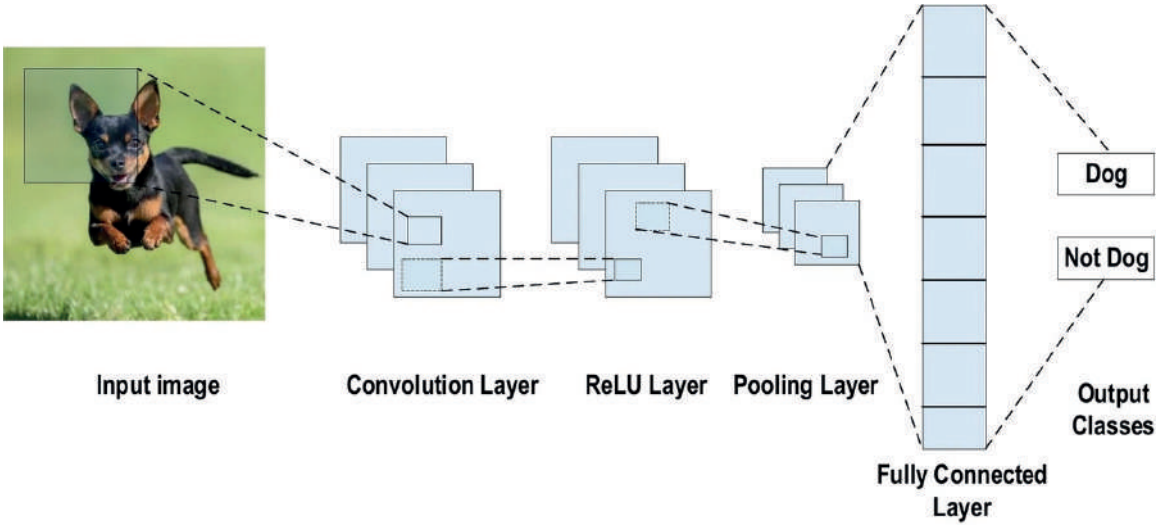


Figure 3.6: This image [10] shows the layers of a convolutional neural network, starting with the convolution layer, followed by the ReLU nonliear activation function, the pooling layer and the fully–connected layer which outputs the image class.

particular aspects, with deep learning overall moving away from its initial goal of modeling the human brain.

The potential of CNNs to be used with images, speech, and time series data has been recognized since 1997 [78]. CNNs have been successfully used in multiple tasks, more notably in computer vision and subsequently natural language processing. CNNs were some of the first models that showed high accuracy for practical and commercial applications, giving more credibility to deep learning. More recent applications of CNNs include high efficiency, flexible networks [26], large–scale video classification models [71], and classification of multi–channel time series data obtained from human activity [146]. However, CNNs have had the largest impact on two–dimensional image data. For one–dimensional, sequence data, part of which are time–series signal data, another neural network framework has been most prominent: recurrent neural networks, described next [45].

3.2.2 Recurrent Neural Networks

Recurrent Neural Networks (RNN) [112], as summarized by Goodfellow [45], are a set of neural networks used to process sequential data, including time series data. Most of the algorithms in this family can scale easily and handle input of variable sizes. Similarly to CNNs, RNNs use parameter

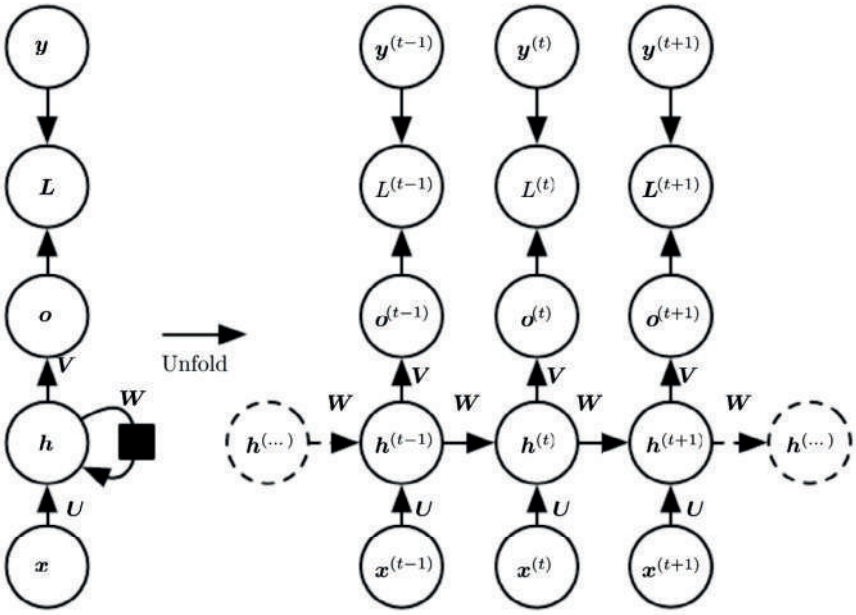


Figure 3.7: This illustration [45] shows the computational graph of a recurrent neural network, with the figure on the right representing the unfolded version of the network on the left over time, or over the sequence steps. The first layer, x, is the input, continuing with several recurrent hidden layers h, then with the output layer o. The subsequent layer L calculates the error between o and the corresponding training target y.

sharing to generalize ideas and decouple parameters from their positions in the sequence. Time series 1D convolution relies on the same parameter sharing property, by applying the same kernel to each time step, but it is shallow and does not have memory. In RNNs, each output node is a function of a previous one, produced using the same update rule, applied to the previous outputs. RNNs are applied to a sequence, with the time step not necessarily indicating time in the real world, but the position of the value in the sequence. RNNs may also be applied to 2D data. RNNs include cycles which allow one variable in the present to influence its own value in the future. This is accomplished by, for each step, allowing previous outputs to be utilized as inputs in addition to hidden states of traditional MLPs. Additionally, RNNs allow an output of variable length, also enabling sequence–to–sequence conversions, particularly useful for tasks like machine translation or time series forecasting. A typical RNN architecture is illustrated in fig. 3.7.

This flexibility makes these architectures ideal for classification of input of any length, while maintaining limits on model size, taking into account historical information, and sharing weights

across time. A known drawback of RNN models is their difficulty in capturing long–term dependencies due to multiplicative gradient. The error of such models can be exponentially decreasing or increasing with respect to the number of layers. The Gated Recurrent Unit (GRU) [25] and their generalizations, such as Long Short–Term Memory units (LSTM) [41], address to some extent the vanishing gradient problem encountered by traditional RNNs. I use LSTMs to model the temporal dependencies in the sensor data.

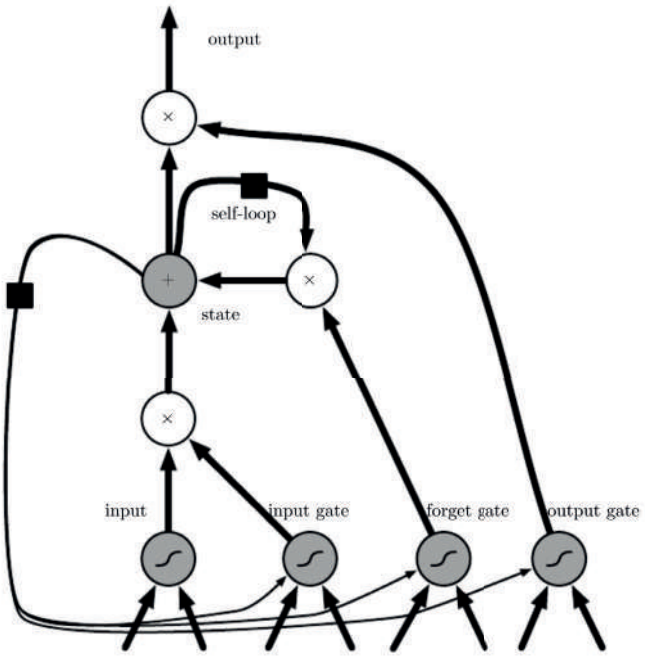


Figure 3.8: Long–Short Term Memory Network Cell [45].

An LSTM is part of a sub–group of RNN algorithms, called gated RNNs. The fundamental idea behind this group is creating paths through time to maintain a gradient that neither vanishes, which would mean losing relevant information from previous states, nor explodes, which would cause that previous information to be given too much importance. This is implemented in gated RNNs through weights that connect nodes which are dynamic and may change during each time step. This features enables the network not only to accumulate information over time, but also forget an old state when no longer necessary. The network learns to decide when to let go of the old state [45].

Figure 3.8 above shows one LSTM network cell. Each cell is recurrently connected to other ones, replacing regular hidden cells of neural networks. In this case, the input value can be incorporated into the state if the input gate allows it. The state unit has a self–loop with a weight adjusted by the forget gate. The output of this cell's computation can be suppressed by the output gate. LSTM

networks are not all identical, and several modifications to this cell have been used. LSTM networks have been demonstrated to be more capable of learning long–term dependencies than simple RNNs.

One of the defining characteristics of LSTM model was the addition of self–loops which allow the gradient to flow and be used for long durations [59], to be followed by making the weights associated with this gradient context–dependent rather than fixed and enabling forgetting [41]. This way, the time scale of integration changes based on the input sequence since the constants are produced by the program itself. LSTM has been very successful in many applications, such as speech recognition [48, 50], handwriting recognition [49] and generation [47], machine translation [126], and image captioning [145].

The algorithms and neural network architectures detailed in this chapter are integral to constructing computational models that enable accurate working of the textile sensors designed with one conductive yarn. The application of such models to these novel pervasive technologies is key to unfolding their potential, particularly when real–world and user experience considerations are also included, an aspect upon which the subsequent section starts to expand.

3.3 Smart Textiles in the Real World

Real–world integration of minimalistically–designed kitted sensors is an important aspect which this work addresses. This section starts by briefly discussing qualitative studies involving smart textiles, followed by a description of how machine learning has been used to enhance other fabric–based applications. These are both areas of interest to this work, which aims to enable interactivity in knitted sensors at the intersection of artificial intelligence and user experience.

3.3.1 Qualitative Studies with Smart Textiles

To properly explore the interaction space of smart textiles, end users' views should also be considered: their overall perceptions, hesitations, and desired applications. Qualitative studies in previous work have started addressing some of these questions but have mainly revolved around specific aspects of a technology or particular applications. The work in [69] has used participatory design methodology to develop applications with toolkits. Other research has explored user preferences for finger gestures on

smartwatches [115], wrist gestures for in-vehicle application control [97], and force input on steering wheels [63].

Participatory design methodologies [123] have been conducted to develop specific applications for new technologies in various fields, such as education [81], medical fields involving experts and physicians [148, 121, 31], and to understand the needs of special groups [131]. These methodologies have also been adopted for e–textiles and fabric–based sensors [96, 69]. One example is the work from Devendorf et al [30], which, through user studies, examines dynamic textile displays in relation to personal style, which is typically associated with complex personal and socio-cultural significance [56, 44]. That work is based on Gaver et al.'s [39] original descriptions of ambiguity as a resource in design. Other work from Davis et al. [28] explored the emotional impact of textiles' textures and materials on users. These studies, however, did not explicitly focus on users' desired application space for touch-sensitive textiles, or their concerns for everyday interaction. The work in this dissertation aims to utilize user perspectives to guide design aspects of future interactions based on touch–sensing fabrics in general, and more particularly, those applications that would rely on fabrics knitted with one conductive yarn and minimal external connections. The study protocol aimed to allow diverse input on key interface aspects of look and feel, accessibility, and intuitiveness [124].

3.3.2 Machine Learning Applications for Fabric–based Touch Sensing

Given the power and wide areas of applicability of machine learning, several smart textiles, broadly discussed in section 2.1.1, have also benefited from it. While textile touch sensors can provide basic touch location, machine learning algorithms have been used to enable higher–level, accurate input recognition. *Project Jacquard* [108] introduces a platform to produce interactive woven sensors. In its user evaluation study, gestures such as *swipe left*, *swipe right*, and *hold* were recognized under three different condition: *sitting*, *standing*, and *walking*. The work from Hughes et al. [64, 65] describes the construction of an RF–based e–textile and its ability to distinguish among gestures of *tap*, *up swipe*, and *down swipe* based on a CNN model. *SmartSleeve* [103], a deformable, pressure–sensing textile sensor recognizes in real time both surface and deformation gestures. It relies on an algorithm which, after converting the data in image form, uses an SVM–trained model together with

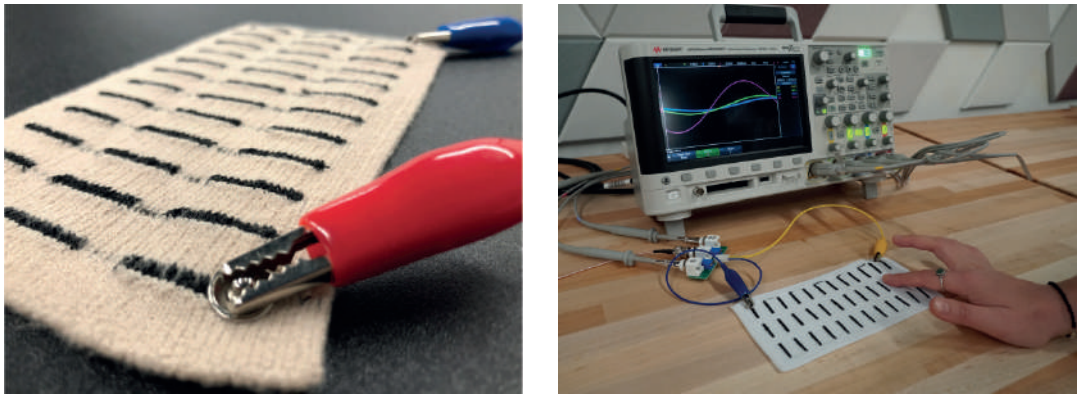
heuristics to detect the gesture type. *GestureSleeve* [114] is a touch–sensing textile sleeve, able to detect stroke–based gestures or taps through the $P algorithm [136]. *Smart–mat* [125] uses a kNN classifier on features built using time series statistical information. *RESi* [101], a resistive, touch sensing interface uses SVM classification to detect basic gestures such as *single, double and triple taps*, as well as *swipes in the four directions*. More recent work [99] furthers cord–based interfaces by enabling continuous control, as well as recognizing casual discrete gestures.

Part of the contribution of this work is detecting various gestures, some of which, mainly described in chapter 7, that are relatively complex compared to most of these examples. Additionally, all the sensors described above, primarily focused on gesture recognition, differ from this work in construction and sensing strategy. In many of these cases and other similar sensors, much of the complexity lies on the hardware design. In the sensors on which this work focuses however, the construction is kept simple and streamlined, while shifting the burden of complexity to computational models, which includes building algorithmic and deep learning recognition models. The next chapter starts tackling that complexity by introducing algorithms to represent the signals generated from these sensors in terms of their most important aspects, and to calculate the similarity between time–series signals.

Chapter 4: Signal Representation

4.1 Introduction

This chapter focuses on signal representation methods to prepare for subsequent work in accurately detecting touch location and gestures on minimalistic knitted sensors. The knitted capacitive sensors [133] described in section 2.1.2, rely on one conductive yarn and only two fabric-to-wire connections at yarn endpoints (fig. 4.1(a)). This design simplifies connection to sensing electronics and improves wearability by making the textile more flexible and durable. However, sensing techniques in [133], which were extensively described in section 2.3.2, only approximate general areas of touch. Therefore, in order to enable the detection of fine touch location, some signal processing and algorithmic contributions are necessary.



(a) (b)

Figure 4.1: Examples of the knitted capacitive touch sensor [3]: *(a)* A planar knitted fabric touchpad made from polyester and carbon-suffused nylon connected to two electrodes through button snap fasteners. *(b)* Oscilloscope visualization of the voltage input and output from the fabric circuit.

First, the use of Bode analysis [18] with *differential capacitive sensing* [133] is introduced to improve the precision of measured data (fig. 4.1(b)). Bode analysis is a system identification approach, used to characterize the behavior of an unknown system given controlled inputs, briefly summarized in section 2.3.2. Furthermore, I introduce *MSD*, an algorithm to extract invariant features from time-series signals, which sparsely represents the captured signal in a more meaningful way regard-

ing touch location identification. *MSD* is based on the scale-space of a signal and on *SIFT*, outlined in section 3.1.1. This method is not used to classify location of touch at this point. Rather, both *MSD* and the application of Bode analysis aim to improve *touch location representation* while maintaining compatibility with existing knitted textile sensors. A 13-subject user study was conducted to evaluate these methods. In order to determine the level of similarity between different samples of touch signal representation, I introduce the *ELD* algorithm. This generalizable distance metric measures the similarity between two tensors of varying lengths.

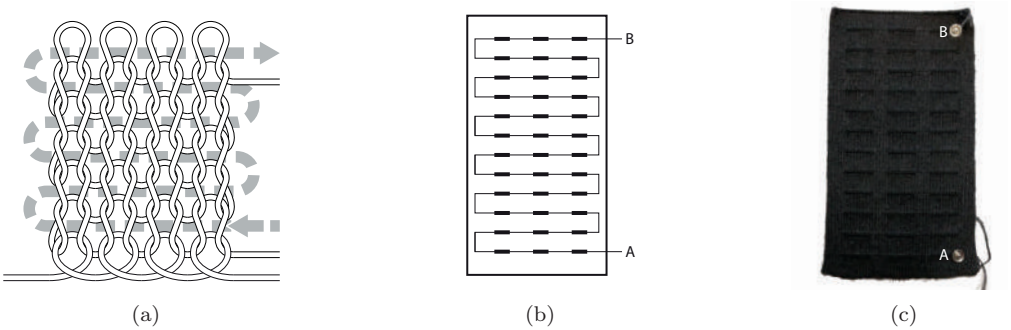
Section 4.2 discusses the application of Bode analysis to *differential capacitive sensing*, which is followed by a description of the signal acquisition pipeline in section 4.3. Section 4.4 describes *MSD*, an algorithm to represent touch input events in terms of sparse, invariant features. Section 4.5 focuses on *ELD*, an algorithm to compute similarity between pairs of sequential data describing touch events. *ELD* is generalizable to tensors of variable lengths, making this algorithm compatible with the representations generated by *MSD* are not uniform in size. Subsequently, the experimental procedure and user study are discussed in section 4.6, followed by results in section 4.7 and conclusion in section 4.8.

4.2 Bode Analysis with DCS

The sensor used in this work [133], illustrated in fig. 4.2(c), is a 4 inch $\times$ 8 inch touchpad with 36 points of contact, or 'buttons", and a cross–knitted resistance of 550 kΩ. The buttons in this design are spaced approximately 2/3 inches apart. The inter–button resistance is approximately linear at 15 kΩ between button center–points.

As discussed above, in addition to the knitted component, prior work [133] describes a knitted capacitive touch circuit that is sensitive to touch across the continuous conductive yarn. A current differential is measured using conventional capacitive sensing techniques that are susceptible to distortion, which limits the overall location resolution. The contributions of signal processing to this work improve the precision and separability of measured data.

Improving the performance of the knitted touch sensor requires resolving fine changes in capacitance in real time. The measurement strategies used by conventional capacitive sensing controllers



(a) (b) (c)

Figure 4.2: Weft knitted touch sensor design [133]. Figure produced at CFF [3].: *(a)* Illustration of the serpentine pathway formed by the yarn during weft knitting. *(b)* Illustration of the conductive yarn pathway knitted during touch sensor construction. *(c)* Image of a complete knitted touch sensor.

do not account for distortion while capturing the voltage waveform. Instead, distortions present within the time–series output are processed using a windowed moving average or convolution to remove high–frequency variations and improve touch threshold detection. Filtering measured data does not improve the overall quality of the data captured and, in the case of measuring the strength of capacitance, distortion within measured data decreases the effective resolution of touch localization.

As an example, we observe the waveforms recorded from input at adjacent touch locations on the yarn separated by approximately 30 kΩ. A 10 kHz sine wave with a peak–to–peak voltage of 5V is input through both current limiting resistors. Voltage is sampled at approximately 8.14 MHz and 815 points per frame for 250 frames. Figure 4.3(a) and fig. 4.3(b) plot the distribution of data measured through phase analysis. Figure 4.3(a) shows data measured with negligible interference, while fig. 4.3(b) shows the effects of measurement distortion from a nearby fluorescent light bulb outputting interference above 40 kHz. The distortion in the time measurements contributes to the larger variance between fig. 4.3(a) and fig. 4.3(b). As we can see, however the touch location data for different touch positions overlaps in both cases, especially under higher–interference conditions.

By applying Bode analysis toward differential capacitive sensing, the goal is to remove the variance present in data measured by conventional capacitive sensing. Fine changes in capacitance are measured by sampling the continuous voltage waveform over time, performing a Fourier transform and extracting the gain ratio from the measurements at the input frequency bin. This process allows

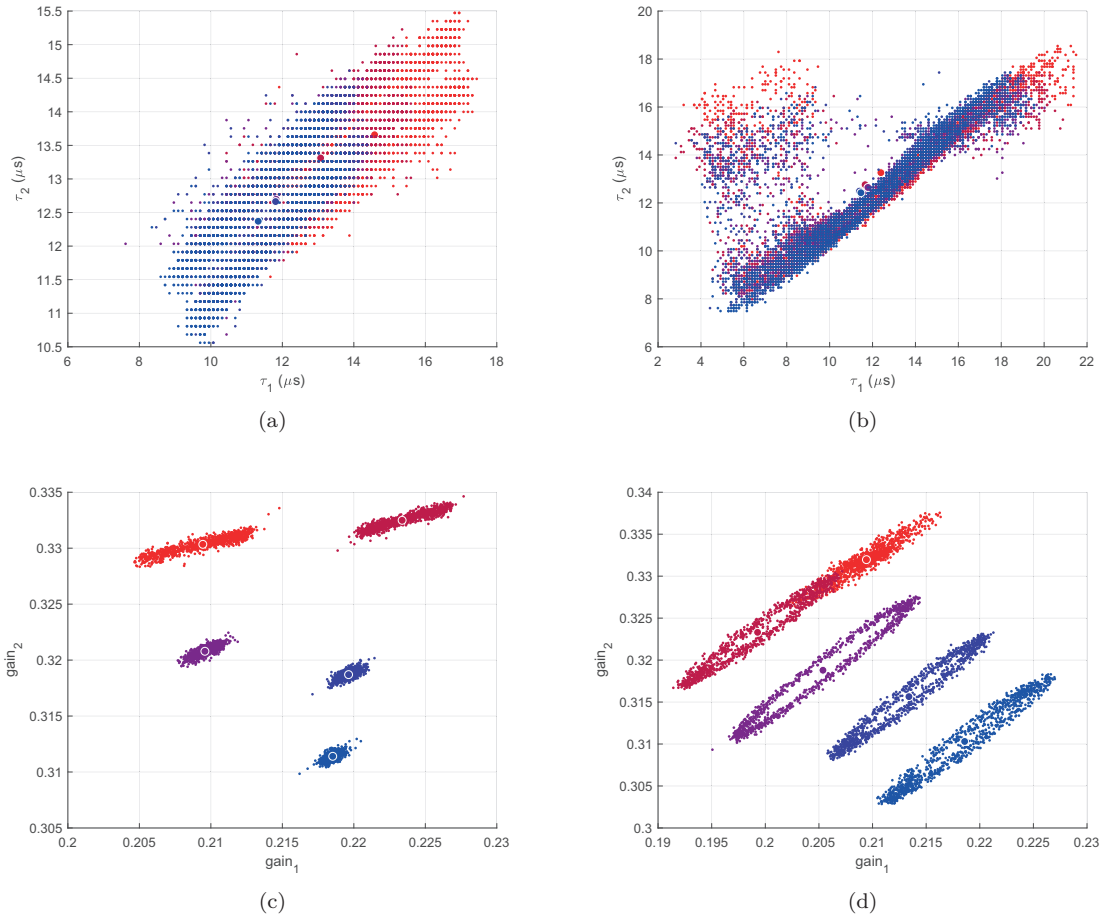


Figure 4.3: Comparison of phase and Bode analysis. Figure produced in collaboration with CFF [3]. *(a)* Phase analysis without EMI distortion. *(b)* Phase analysis with EMI distortion. *(c)* Bode analysis without EMI distortion. *(d)* Bode analysis with EMI distortion.

selectively discarding all other frequencies that are not present within the input. In this example, a single frequency is input. Measuring one full period of the voltage signal yields information at the first frequency bin at 10 kHz. Figure 4.3(c) shows the clustering of processed data without induced noise, while fig. 4.3(d) is produced in the presence of EMI as in the case of fig. 4.3(b). Table 4.1 summarizes these results by calculating the standard deviation among the points for each condition.

Table 4.1: Signal analysis strategy comparison.

	Phase w/o EMI	*Phase w/ EMI*	*Bode w/o EMI*	*Bode w/ EMI*
Standard Deviation	1.24	3.28	0.053	0.056

The clusters of points are easily separable in fig. 4.3(c), and that separability is still retained even

in the presence of noise, since the clusters do not overlap. However, the data still has some spread and not completely accurate localization ability. Section 4.4 describes a sparse, invariant signal representation algorithm, to further move toward accurate localization. The section below provides more details about the capacitive signal acquisition process and its post–processing techniques.

4.3 Signal Acquisition

The signal processing pipeline, illustrated in fig. 4.4, describes the process of generating an excitation waveform, measuring the direct output from the textile sensor and converting sensed data into amplitude gains for the chosen frequency.

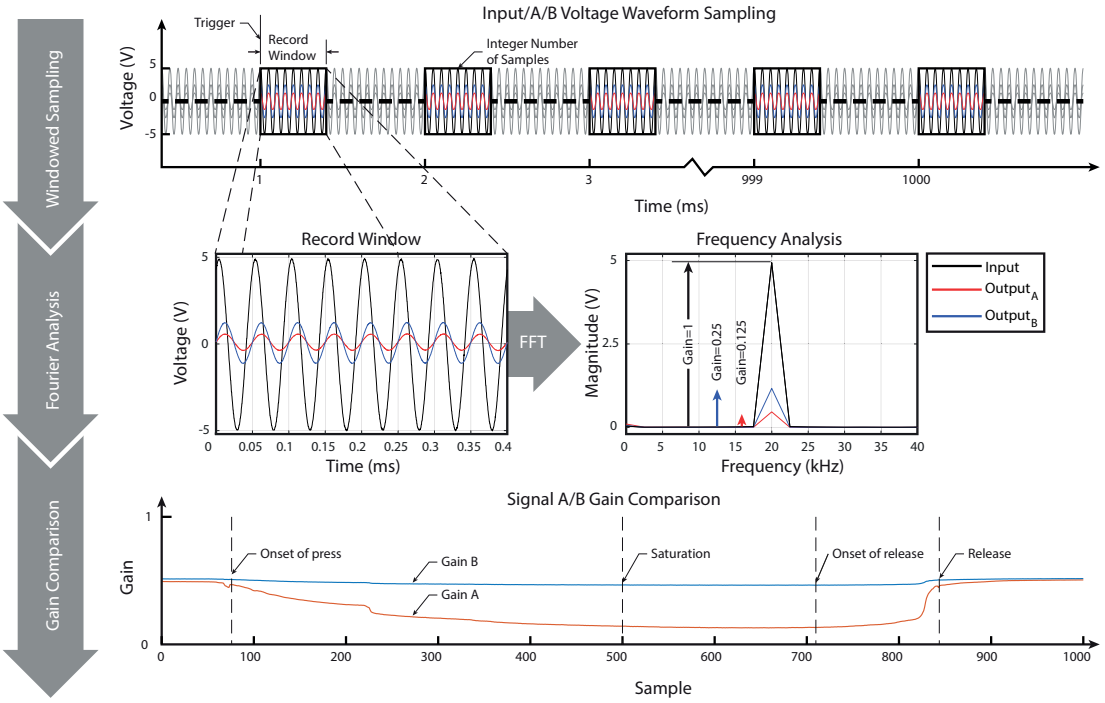


Figure 4.4: Pipeline for waveform analysis. Figure produced at CFF [3]. The input and output waveforms are recorded in segments which are individually processed. The spectrogram of the record window is computed to find the single–sided amplitude of the input and outputs. The ratio of input to output is computed as the output gain.

4.3.1 Waveform Capturing

The voltage waveform data is recorded using a Keysight MSOX–3024A 4–channel mixed–signal oscilloscope. Channel 1 measures the input voltage waveform. Channels 2 and 3 measure the A and

B voltage outputs. Channel 4 measures the sample trigger. A Keysight 33622A function generator is used to generate both the voltage input and the sample trigger. A 20 KHz sine wave is input at both fabric endpoints. The measurement window spans 400 μs with 1 ms between sample window triggering. Each window captures 8 periods of the input and output waveforms at approximately 4.06 MHz (1627 points per waveform). During each touch event, 250 sample windows are recorded over the duration of 2 seconds.

4.3.2 Post–Processing

Fourier analysis of each sample window is conducted during the second phase of the pipeline to determine the voltage magnitude present at the given input frequency. Because an exact integer number of periods (8 periods) is sampled within the window, the output of the Fourier transform will the be present at the 8^{th} frequency bin. In embedded applications, this detail simplifies analysis of the waveform by placing all of the relevant data at an exact bin location.

The magnitude values of the input and two outputs are used to compute the normalized voltage gain of outputs A and B in the third phase of the pipeline. $Gain_A$ and $Gain_B$ are computed as $Output_A/Input$ and $Output_B/Input$. The plot in the third phase shows the attenuation during a press–and–release event on the left side of the sensor. The dip in gain is much greater towards the sensing electrode at A than at B.

While it is possible to use analytical heuristics to decouple the touch pressure and touch position, I propose a more robust decoupling methods, detailed below, which is then compared to the signal data processed only as described in this section.

4.4 Invariant Feature Construction

As described above, two independent waveforms are recorded from the sensor–one at each end of the conductive yarn. Features are then built based on the gains of the voltages levels from each signal measured over time. The voltage gains are rendered using Bode analysis and saved as a 250×2 matrix of time–series data. My proposed approach, *Mixed–Source Description*, produces a sparse representation of binary signal time–series data, aiming to capture the relevant information and

discard noise. *MSD* is based on Bag–of–Temporal–SIFT–Words (BoTSW), which detects invariances in the scale–space of each signal, and is adapted to incorporate a joint description of the two waveforms.

The scale–space is a multi–scale representation of a signal. Scale- -space enables the detection of structures at different scales, where a new scaled version of the signal is achieved by smoothing the original one. At each scale, its salient features are extracted as key-points and then locally described. The BoTSW adapts SIFT to 1–dimensional time series signals instead of 2–dimensional images, for which SIFT was originally developed, while also incorporating the concept of Bag–of–Words from natural language processing. *MSD* does not use that concept in its representation. An overview of SIFT is provided in section 3.1.1.

4.4.1 *MSD* Algorithm Steps

All the steps of this approach are further explained below. The algorithm workflow is illustrated in fig. 4.5 and fig. 4.6. The key-point detection step in fig. 4.5 includes one of the main differences between *MSD* and BoTSW: if a key–point is detected in one of the signals, its associated point in the other signal is assigned as a key-point as well. Figure 4.6 illustrates another difference with BoTSW, with descriptors for each key–point being normalized, to be subsequently concatenated. Both SIFT and BoTSW, and as a result *MSD*, rely on some relatively well–known concepts such as Gaussian scale-space and Difference of Gaussian. A brief overview of these concepts is provided in section 4.4.1 for clarity and completeness. For more details on the existing algorithms, please refer to [90] and [16].

Gaussian Blurring

Each of the two 1D signals present in one key–press were blurred separately in five different scales (σ) using a Gaussian kernel $G(t, \sigma)$ [87, 84, 14] of size 250 and standard deviation σ, corresponding to the size of the original signal interval. Blurring refers to the operation of performing convolution of the original time series signal interval $S(t)$ with the Gaussian operator, to produce $L(t, \sigma)$ as

shown in equation (4.1).

$$L(t, \sigma) = G(t, \sigma) * S(t) \tag{4.1}$$

The scales selected for our experiments follow the suggestions of the original SIFT algorithm [90] and differ from each–other by a factor of $k = \sqrt{2}$, starting with $\sigma = 0.5$.

Difference of Gaussian

In order to find the areas of rapid change in the signal, the Laplacian, which is the second order derivative of the functions, needs to be computed on the blurred signal. Since the Laplacian is sensitive to noise, smoothing the signal through blurring helps stabilize its representation. In order to preserve scale–invariance, the Laplacian of Gaussian needs to be scaled by σ^2. Computing the Laplacian is expensive–however, it can be closely approximated by the Difference of Gaussian (DoG) operation [85, 90]. The DoG at a particular scale can be computed by subtracting blurred signals of subsequent scales from each–other, as shown in (4.2).

$$D(t, \sigma) = L(t, k_{sc}\sigma) - L(t, \sigma). \tag{4.2}$$

A signal filtered with a Gaussian of the scale of DoG being computed is subtracted from that same signal filtered with a scale k_{sc} times greater, which is the next scale up, as shown in fig. 4.5 (b). Since we blur the signal in five different scales, there are four DoG signals produced for each time–series measurement.

Key-point Extraction

Key-points are discovered from the DoG representations at a particular scale σ. Specifically, a key–point is defined as an extremum, which means that that point is the maximum or the minimum of the points in its neighbourhood. A point's neighbourhood is comprised of points in its own scaled DoG representation, $D(t, k_{sc}\sigma)$, as well as points in the DoG representations of one scale below $D(t, k_{sc-1}\sigma)$ and one above $D(t, k_{sc+1}\sigma)$.

For each 250–instance interval corresponding to a key–press, every point in the scale–space of

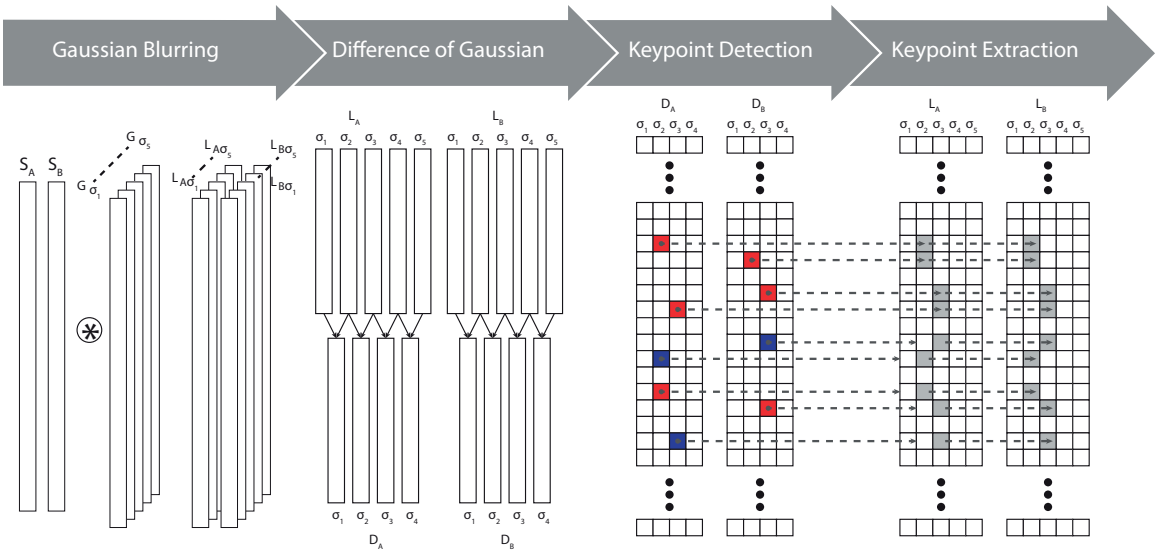


Figure 4.5: Invariant feature construction process for each interval of two signals. Figure produced in collaboration with CFF [3]. (a) : Gaussian blurring–individual kernel application to each sensor data to produce filtered signals in different scales (b) : Difference of Gaussian–computed by subtracting pairs of consecutive filtered signal data (c_i) : Key-point detection–checking the DoG signals for points that are local maximums or minimums (c_{ii}): Key–point extraction–Finding the key-point according to its co-ordinates of position t, and scale σ in the filtered signal pair, where the point of the same co–ordinates in the other signal is considered a key–point as well, and extracted.

each of its 1D signals is compared to its neighbourhood. If the point in the DoG representation is detected to be a key–point, its position t and scale σ are retained. The neighbourhood of each point is composed of one point before and one point after itself in its own scale, as well as the three corresponding points to the previously mentioned ones in the scale above, and the scale below. In total, each point of every DoG is compared to eight other points. The touch location dataset is composed of two signals measured simultaneously, and each signal interval's DoG is searched for key-points separately, as in BoTSW. If a key-point is discovered in one, the corresponding position t and scale $k_{sc}\sigma$ are retained to be subsequently described.

Descriptors

After key–points are detected, each is used as a coordinate of position t and scale $k_{sc}\sigma$ to a point in the filtered signal representation $L(t, k_{sc}\sigma)$. That point located in the filtered signal is then described in terms of a fixed number of other points that surround it in that scale space.

Blocks are defined by grouping some of these points together. For the key-point descriptors, I

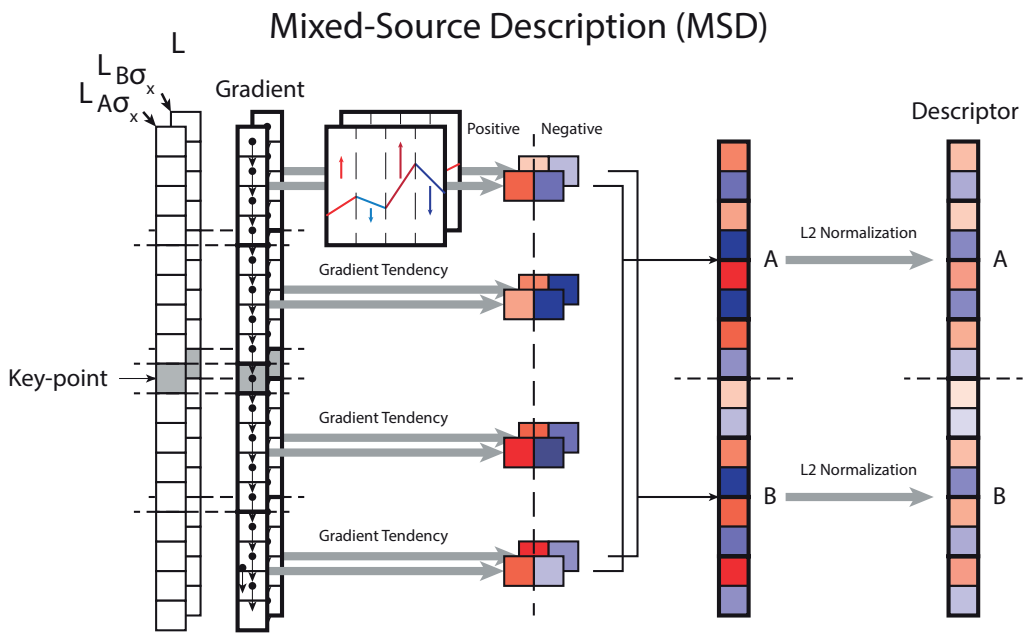


Figure 4.6: Construction of *MSD* descriptors. Figure produced in collaboration with CFF [3].

use a total of 4 blocks of 4 points each, with two blocks sequentially before the key-point and two blocks after. For a 1D signal, the neighbourhood consists of a total of 17 points: 2 blocks of 4 points each before the point, the point itself, and 2 blocks of 4 points after the point. Next, gradients are computed for every point of the 17×1 vector, to produce another vector of the same size. Within each block of the gradient interval, positive gradients are summed together, as are negative ones. These two values, the sum of its positive gradients and the sum of its negative ones, are used to describe one block. A point is represented by the descriptors of all its blocks, resulting in a feature vector whose size is double the number of blocks: 8 in this case.

MSD Descriptors

BoTSW uses one time series signal over which it extracts key–points and computes descriptors as described above. I apply the same principle and, additionally, use the fact that there are two simultaneous signals for each key–press to capture inter–dependencies between them. If a key–point is discovered in one of the signal's filtered representations, L_σ, the corresponding point is locally described within that 1D signal. In the *MSD* algorithm, additionally, the point of same position

t and scale σ in the other signal is considered a key–point by association, even if it has not been detected as an extremum. That key–point is also described the same way within its own signal column. Next, the descriptors of these associated key-points of the same t and σ, but different capacitive sensor sources, are each individually normalized using *L2–normalization* to increase the stability and robustness of representation. They are then concatenated to create a 16D feature vector. The computing of *MSD* descriptors is shown in Figure 4.6. In BoTSW [16], each key–point is described only within its neighbourhood–since there is no other coupled signal from which to define an associated key–point, and it is not normalized.

The reason for joint description in *MSD* is the fact that the two signals are related, since they are produced by the same touch input measured at either end of the conductive yarn. Therefore, if an invariance is detected in one of these signals, the state of the other at that point in time becomes important as well. The produced descriptor in our case contains cumulative information from two different waveforms, but this method is extensible to signals from multiple sources related in time.

As a result of *MSD* computation, a key–press is represented by a sequence of descriptors. The size of each descriptor is fixed for every key–press, and it is greater than the size of any single sample from the original key–press by a factor of 8. The number of descriptors per key–press varies, since it depends on the number of detected key–points for that key–press. In any case, the *MSD* representation of a key–press is sparser than that of the original signal, resulting in a shorter sequence.

Codebook Generation

BoTSW contains the extra step of *codebook generation* in feature construction, which is omitted in *MSD*, so the quality of generated descriptors can be better analyzed. Codebook generation consists of first clustering all key–point descriptors of the data set using the K-Means algorithm, then calculating for each interval the distribution of its key–points over the clusters. This distribution would constitute the feature vector of each interval. Thus, a matrix of dimension 250×2 would be represented by one vector of size equal to the number of clusters selected for clustering. While this approach would still give insight into the data-set, the number of samples to analyze would be

considerably reduced. Moreover, as is the case with K–Means clustering, the number of clusters would need to be selected experimentally, without any particular reason applicable to this type of problem. This would obscure the answer to the primary question of whether key–points described in terms of their neighborhood are a good choice for feature construction.

4.4.2 Algorithm Summary

The purpose of the *MSD* algorithm is to represent a two–channel signal in a way that is meaningful in terms of its most salient features. It is based on the concept of scale–space, which enables capturing relevant changes in a signal's behavior over different scales. Subsequently, each change is described in terms of its context, the surrounding neighbourhood of points, and the signal itself is represented as a sequence of such descriptors. In section 4.6, the signal representation of *MSD* is compared to that of the raw signal in terms of their abilities to distinguish between key-press events of the same key, and key-press events of different keys. However, before doing that, in section 4.5, I introduce *ELD*, a distance metric to compute the similarity of pairs of key–presses, generalizable to computing distances of tensors of varying lengths. The experiments in section 4.6 show that the proposed sensing method, described in section 4.2 and section 4.3 results in high–fidelity signals. Furthermore, the sparse representation of key–presses produced by *MSD* significantly increases separability between different touch locations.

4.5 Euclidean Levenshtein Distance (*ELD*)

In order to determine the representation power of this sensing method and signal representation algorithm, I designed an experiment where key–press data from different users was collected and compared. A favorable outcome would mean that key–presses of the same location, independent of the user, would be similar to each–other and different from key–presses originating from different touch locations. To that end, in addition, the distance metric introduced below is used to compute the similarity of two key–press events. Key–presses are a multi–dimensional time–series signals of potentially varying lengths, especially if they are represented as key–point descriptors, according to the *MSD* algorithm, described in section 4.4.

In order to measure the similarity between representations of key–presses, I modified the Levenshtein Distance [82], a similarity metric used to measure the difference between two strings. In this work, it is used to investigate how different key–presses that belong to the same button, which is a position along the sensing yarn on the knitted pad, relate to each–other and to key–presses of other buttons.

4.5.1 Definition

This modified version of Levenshtein Distance uses a similar concept to the original in that it performs a pairwise comparison of two sequences, in this case, consisting of key–point descriptors in a key–press. Within a key–press, the discovered key–points, whose order is preserved, are temporally related. Therefore, the list of descriptors on these key–points can be treated as a sequence, since the voltage discharge during a touch event is a process with a relatively pre–defined trajectory. This makes the discovered key–points, and by association, their descriptors, a good fit to be matched to each–other, similarly to characters in a string. In strings, two sequences of varying lengths are aligned to become identical with the smallest number of added, deleted, or transformed characters. Similarly to ERP [23, 22], *ELD* borrows the concept of the *gap* from the Levenshtein Distance. A *gap* character in one string represents the deletion of a character in the other string, in the process of aligning both strings. In the case of *ELD*, the *gap* character is represented by the zero vector $\vec{0}$, enabling the handling of local time shifting by using the $L2$ norm as penalty.

Instead of using a binary 0 or 1 cost to denote a match or mismatch, as in the Levenshtein Distance with string characters, the Euclidean distance is computed between two key–point descriptors—each belonging to one of the key–presses in the pairwise comparison. The total cost of converting a key–press K up to its descriptor $\vec{k_i}$, to the other key–press L up to its descriptor $\vec{l_j}$, is computed by adding the distance between $\vec{k_i}$ and $\vec{l_j}$, or the vanishing costs of $\vec{k_i}$ or $\vec{l_j}$ to the minimum value of the previous step, according to equation 4.4. At every step of the algorithm, according to equation 4.3 either one of the two key–points is kept and the other is discarded, or one is transformed to the other—whichever action has the minimum cost. This process is analogous to deletion, insertion, or character conversion when comparing two strings.

$$dist_{eld}(\vec{k_i}, \vec{l_j}) = \begin{cases} \|\vec{k_i}\|_2 & \text{if } \vec{l_j} \text{ is vanished} \\ \|\vec{l_j}\|_2 & \text{if } \vec{k_i} \text{ is vanished} \\ \|\vec{k_i} - \vec{l_j}\|_2 & \text{if either } \vec{k_i} \text{ or } \vec{l_j} \text{ is converted to the other} \end{cases} \tag{4.3}$$

$$ELD(K, L) = \begin{cases} \max(\|\vec{k_i}\|_2, \|\vec{l_j}\|_2) & \text{if } min(i, j) = 0 \\ \min \begin{cases} ELD(Rest(K), L) + dist_{eld}(\vec{k_i}, gap) \\ ELD(K, Rest(L)) + dist_{eld}(\vec{l_j}, gap) \\ ELD(Rest(K), Rest(L)) + dist_{eld}(\vec{k_i}, \vec{l_j}) \end{cases} & \text{otherwise.} \end{cases} \tag{4.4}$$

In the minimum clause of equation 4.4, from K to L, the first case corresponds to deletion of the current key–point $\vec{k_i}$, the second to insertion, and the third to conversion from $\vec{k_i}$ to $\vec{l_j}$. If two key–presses are compared sample–by–sample, the resulting Euclidean Levenshtein Distance is a measure of similarity between them. A smaller distance indicates higher levels of similarity, since converting one key–press into the other would cost less. While this algorithm was specifically designed to measure the distance of two key–presses, each represented as a sequence of key–points, its uses are not limited to this particular application and can be extended to any problem that aims to measure the minimum cost of converting one tensor to another, particularly useful for multi–dimensional time–series signal analysis. For clarity, section 4.5.3 illustrates a full example of computing the distance between two tensors. First, however a brief proof that ELD is a distance metric is provided below.

4.5.2 Proof of Distance Metric

The required conditions necessary for a function to be a metric are provided in definition 4.5.1. The Levenshtein Distance has been proven to be a metric distance function [139, 22]. However, other distance measures, which are closely related to it, but have been adapted for time–series data,

such as LCSS [138], DTW [17], and EDR [24], are not metric functions, since they violate *triangle inequality* [22].

The proof that *ELD* satisfies triangle inequality is similar to the proof of ERP by Chen et al. [23, 22], and relies on lemma 4.5.1. It should be noted that all operations have non–negative lengths since $dist_{eld}(\vec{a_i}, \vec{b_j})$ is defined in terms of the $L2$ norm, which is a metric.

Definition 4.5.1. A metric on a set X is a function d such that $d : X \times X \to [0, \infty)$. In order for d to be a metric on X, $\forall x, y, z \in X$, the following axioms need to be true:

1. identity of indiscernibles: $d(x, y) = 0 \Leftrightarrow x = y$
2. symmetry: $d(x, y) = d(y, x)$
3. triangle inequality: $d(x, y) \leq d(x, z) + d(z, y)$

Lemma 4.5.1. *For any three vectors $\vec{a_i}$, $\vec{b_j}$, and $\vec{c_k}$, each part of a distinct time series sequence, the triangle inequality needs to hold: $dist_{eld}(\vec{a_i}, \vec{b_j}) \leq dist_{eld}(\vec{a_i}, \vec{c_k}) + dist_{eld}(\vec{c_k}, \vec{b_j})$.*

Proof. According to eq. (4.3), the distance function in all three clauses is defined based on the $L2$ norm of an element of the sequence. The triangle inequality in this lemma holds, since the $L2$ norm, which determines a metric, satisfies triangle inequality. □

Theorem 4.5.2. *Assuming K, L, and M are distinct time–series sequences of varying lengths, the following equation needs to be true: $ELD(K, L) \leq ELD(K, M) + ELD(M, L)$.*

Proof. As in the case of Chen et al. [23, 22], proving this theorem relies on results from the proof by Waterman et al. [139] regarding the Levenshtein distance, and lemma 4.5.1. The Waterman proof demonstrated that the triangle inequality is preserved when defining the distance between two strings based on their best alignment, as long as the triangle inequality also holds for the underlying distance function, which was proved by lemma 4.5.1. □

4.5.3 Example

Let us assume we have two tensors, **A** and **B**, represented as matrices, and we want to calculate their *ELD*. We can consider each as sequences of their rows, where each row has the same number

of elements–in this case 4. We will refer to each element of these sequences as $\mathbf{a}_x$ and $\mathbf{b}_z$.

$$\mathbf{A}_{x,y} = \begin{pmatrix} 1 & 3 & 0 & 2 \\ 4 & 11 & 2 & 3 \\ 7 & 1 & 10 & 0 \\ 5 & 2 & 6 & 8 \end{pmatrix} \qquad \mathbf{B}_{z,w} = \begin{pmatrix} 2 & 1 & 5 & 12 \\ 3 & 4 & 9 & 1 \\ 19 & 7 & 2 & 6 \end{pmatrix}$$

We then define matrix $\mathbf{M}$, which is originally an all zero matrix, and will progressively be populated based on the values of the sequences'elements according to equation 4.4. The size of this matrix depends on the number of elements of the two sequences being compared, and in this example is *4 x 3*.

$$\mathbf{M}_{i,j} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

In order to compute $\mathbf{M}$, we need to rely on the L_2 norms of $\mathbf{a}_x$ and $\mathbf{b}_z$ and in certain instances, the norm of these two vectors'difference. Starting step–by–step:

When $i = 0, j = 0$:

$$\mathbf{M}_{0,0} = max(\|\mathbf{a}_0\|, \|\mathbf{b}_0\|) = max(3.74, 13.19) = 13.19$$

When $i = 0, j = 1$:

$$\mathbf{M}_{0,1} = max(\|\mathbf{a}_0\|, \|\mathbf{b}_1\|) = max(3.74, 10.34) = 10.34$$

When $i = 0, j = 2$:

$$\mathbf{M}_{0,2} = max(\|\mathbf{a}_0\|, \|\mathbf{b}_2\|) = max(3.74, 21.21) = 13.75$$

At this point, the state of $\mathbf{M}$ is the following:

$$\mathbf{M}_{i,j} = \begin{pmatrix} 13.19 & 10.34 & 21.21 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

When $i = 1, j = 0$:

$$\mathbf{M}_{1,0} = max(\|\mathbf{a}_1\|, \|\mathbf{b}_0\|) = max(12.25, 13.19) = 13.19$$

When $i = 1, j = 1$:

$$up = \mathbf{M}_{i-1,j} + \|\mathbf{a}_i\| = \mathbf{M}_{0,1} + \|\mathbf{a}_1\| = 10.34 + 12.25 = 22.59$$

$$left = \mathbf{M}_{i,j-1} + \|\mathbf{b}_j\| = \mathbf{M}_{1,0} + \|\mathbf{b}_1\| = 13.19 + 10.34 = 23.53$$

$$diag = \mathbf{M}_{i-1,j-1} + \|\mathbf{a}_i - \mathbf{b}_j\| = \mathbf{M}_{0,0} + \|\mathbf{a}_1 - \mathbf{b}_1\| = 13.19 + 10.15 = 23.34$$

$$\mathbf{M}_{1,1} = min(left, up, diag) = 22.59$$

When $i = 1, j = 2$:

$$up = \mathbf{M}_{i-1,j} + \|\mathbf{a}_i\| = \mathbf{M}_{0,2} + \|\mathbf{a}_1\| = 21.21 + 12.25 = 33.46$$

$$left = \mathbf{M}_{i,j-1} + \|\mathbf{b}_j\| = \mathbf{M}_{1,1} + \|\mathbf{b}_2\| = 22.59 + 21.21 = 43.80$$

$$diag = \mathbf{M}_{i-1,j-1} + \|\mathbf{a}_i - \mathbf{b}_j\| = \mathbf{M}_{0,1} + \|\mathbf{a}_1 - -\mathbf{b}_2\| = 10.34 + 15.81 = 26.15$$

$$\mathbf{M}_{1,2} = min(left, up, diag) = 26.15$$

At this point, the state of $\mathbf{M}$ is the following:

$$\mathbf{M}_{i,j} = \begin{pmatrix} 13.19 & 10.34 & 21.21 \\ 13.19 & 22.59 & 26.15 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

When $i = 2, j = 0$:

$$\mathbf{M}_{2,0} = max(\|\mathbf{a}_2\|, \|\mathbf{b}_0\|) = max(12.25, 13.19) = 13.19$$

When $i = 2, j = 1$:

$$up = \mathbf{M}_{i-1,j} + \|\mathbf{a}_i\| = \mathbf{M}_{1,1} + \|\mathbf{a}_2\| = 22.59 + 12.25 = 34.84$$

$$left = \mathbf{M}_{i,j-1} + \|\mathbf{b}_j\| = \mathbf{M}_{2,0} + \|\mathbf{b}_1\| = 13.19 + 10.34 = 23.53$$

$$diag = \mathbf{M}_{i-1,j-1} + \|\mathbf{a}_i - \mathbf{b}_j\| = \mathbf{M}_{1,0} + \|\mathbf{a}_2 - \mathbf{b}_1\| = 13.19 + 5.20 = 18.39$$

$$\mathbf{M}_{2,1} = min(left, up, diag) = 18.39$$

When $i = 2, j = 2$:

$$up = \mathbf{M}_{i-1,j} + \|\mathbf{a}_i\| = \mathbf{M}_{1,2} + \|\mathbf{a}_2\| = 26.15 + 12.25 = 38.40$$

$$left = \mathbf{M}_{i,j-1} + \|\mathbf{b}_j\| = \mathbf{M}_{2,1} + \|\mathbf{b}_2\| = 18.39 + 21.21 = 39.60$$

$$diag = \mathbf{M}_{i-1,j-1} + \|\mathbf{a}_i - \mathbf{b}_j\| = \mathbf{M}_{1,1} + \|\mathbf{a}_2 - \mathbf{b}_2\| = 22.59 + 16.73 = 39.32$$

$$\mathbf{M}_{2,2} = min(left, up, diag) = 38.40$$

At this point, the state of $\mathbf{M}$ is the following:

$$\mathbf{M}_{i,j} = \begin{pmatrix} 13.19 & 10.34 & 21.21 \\ 13.19 & 22.59 & 26.15 \\ 13.19 & 18.39 & 38.40 \\ 0 & 0 & 0 \end{pmatrix}$$

When $i = 3, j = 0$:

$$\mathbf{M}_{3,0} = max(\|\mathbf{a}_3\|, \|\mathbf{b}_0\|) = max(11.36, 13.19) = 13.19$$

When $i = 3, j = 1$:

$$up = \mathbf{M}_{i-1,j} + \|\mathbf{a}_i\| = \mathbf{M}_{2,1} + \|\mathbf{a}_3\| = 18.39 + 11.36 = 29.75$$

$$left = \mathbf{M}_{i,j-1} + \|\mathbf{b}_j\| = \mathbf{M}_{3,0} + \|\mathbf{b}_1\| = 13.19 + 10.34 = 23.53$$

$$diag = \mathbf{M}_{i-1,j-1} + \|\mathbf{a}_i - \mathbf{b}_j\| = \mathbf{M}_{2,0} + \|\mathbf{a}_3 - \mathbf{b}_1\| = 13.19 + 8.12 = 21.31$$

$$\mathbf{M}_{3,1} = min(left, up, diag) = 21.31$$

When $i = 3, j = 2$:

$$up = \mathbf{M}_{i-1,j} + \|\mathbf{a}_i\| = \mathbf{M}_{2,2} + \|\mathbf{a}_3\| = 38.40 + 11.36 = 49.76$$

$$left = \mathbf{M}_{i,j-1} + \|\mathbf{b}_j\| = \mathbf{M}_{3,1} + \|\mathbf{b}_2\| = 21.31 + 21.21 = 42.53$$

$$diag = \mathbf{M}_{i-1,j-1} + \|\mathbf{a}_i - \mathbf{b}_j\| = \mathbf{M}_{2,1} + \|\mathbf{a}_3 - \mathbf{b}_2\| = 18.39 + 15.52 = 33.91$$

$$\mathbf{M}_{3,2} = min(left, up, diag) = 31.91$$

Finally, the state of $\mathbf{M}$ is:

$$\mathbf{M}_{i,j} = \begin{pmatrix} 13.19 & 10.34 & 21.21 \\ 13.19 & 22.59 & 26.15 \\ 13.19 & 18.39 & 38.40 \\ 13.19 & 21.31 & 33.91 \end{pmatrix}$$

As a result, $ELD(\mathbf{A}, \mathbf{B}) = 33.91$.

4.6 Experimental Evaluation

This experiment was conducted to test the validity of the capacitive sensing and signal representation methods. In a user study, 13 participants briefly pressed each of the 36 discrete touch locations on the knitted capacitive sensor pad. The contiguity of touch positions along the conductive yarn pathway poses challenges to accurately localizing discrete touch events. The added difficulty of this task is that time–series signals, especially those that are prone to noise, are difficult to represent in a meaningful way.

4.6.1 Experimental Questions

The aim of the methods described in this chapter is to capture a signal from the sensor which is representative of a particular touch location. Furthermore, it would be beneficial to reduce some of the complexity of the problem by retaining information from that signal which encodes meaningful underlying changes between touch positions and disregards similarities. The approach introduced in this work is not intended to substitute automatic feature extraction, but rather to enhance it as a pre–processing step to a more advanced neural network which can further learn to distinguish between different positions–but from a more informed starting point.

The focus of these experiments is to test whether:

1. The data obtained from the *differential capacitive sensing* strategy used with Bode analysis provides statistically significant separability between locations of key–presses.

2. The proposed representation, produced by *MSD*, increases intra–class similarity and decreases inter–class similarity, compared to that of the raw signal data.

ELD, described in section 4.5, is used as a distance metric in both cases, since this algorithm computes distances between multi–dimensional time–series sequences of varying lengths.

4.6.2 Experimental Procedure

A 36–button touchpad with the same design as the one illustrated in fig. 4.2(c) was used for experiments. A group of 13 participants was selected to conduct data collection. The participants' ages ranged from 19 to 45 with an average age of 26. All participants were of good health and exhibited no skin–related medical conditions. Each of the 13 subjects conducted between 10 to 20 independent trials. A trial consists of one subject individually pressing and releasing each of 36 touch points over a duration of two seconds. The data was collected during one or more sittings per subject in a laboratory environment and processed as described in section 4.3. Additionally, a baseline (no–touch) measurement was recorded at the beginning of each trial.

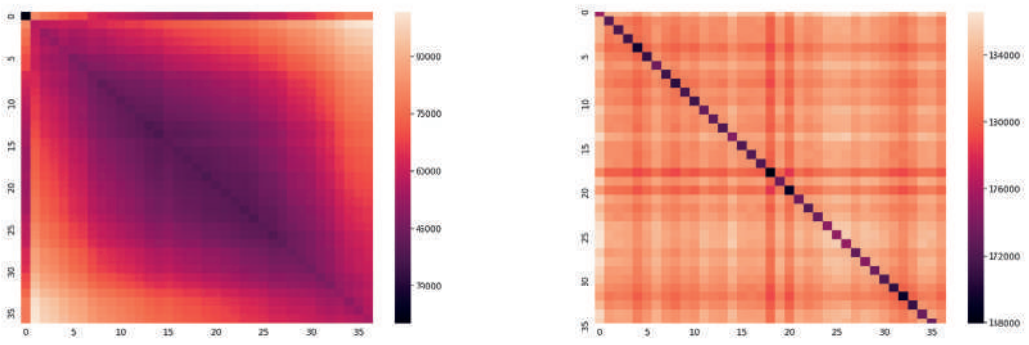
A press on these touch positions, or *buttons*, was similar in duration to a key–press on a keyboard, allowing enough time to capture the onset of press, pressure saturation, release and return to baseline. In order to investigate the inherent behavior of the sensor, the data was collected under ambient environmental conditions without adding further variability to it, such as altering humidity, body poses, etc. Particularly, data collected during these trials was not subjected to interference from fluorescent light or other intense EMI sources. Some subjects did use different fingers when pressing touch positions, however that was not a controlled condition.

A total of 210 key–presses per button were collected from this study. Each key–press is expressed as a matrix of 250 $\times$ 2 gain samples, after being processed as described section 4.3. This matrix representation of key–press data contains 'raw" signal data, serving as the baseline model. It is used to investigate experimental question *(1)*, posed above. In order to explore experimental question *(2)*, the raw signal data was subsequently processed using *MSD*, the algorithm introduced in section 4.4 and illustrated in fig. 4.5 and fig. 4.6. A processed data set of 7560 key–presses was produced, with each key–press containing a variable number of descriptors, depending on the number of key–points

discovered in it. The dimensionality of the *MSD* descriptors is 16.

In order to measure the similarity relationship of different key–presses belonging to the same position, as compared to key–presses that belong to different positions, I compute the pairwise *Euclidean Levenshtein Distance* for every key–press pair in the data set, as described in section 4.5.1. This generates a 36 × 36 matrix, where both rows and columns indicate the number of the touch location on the pad, which is the label of each key–press. Each key–press is associated with its label, and when two key–presses are compared to each other, a scalar value that represents their *ELD* is produced. That value is added to the matrix in position (m,n), where m and n denote the label of the first and second key–presses being compared, respectively.

4.7 Results and Discussion



(a) Raw signal data.

(b) *MSD* signal representation.

Figure 4.7: Heatmaps indicating touch position distances, computed using the sum of *Euclidean Levenshtein Distance* of all key–presses' pairwise comparisons. Each heatmap has a size of 36 × 36, where each cell in either direction corresponds to a touch position. The key–press representations upon which these values were computed was obtained from the raw signal data in (*a*) and the *MSD* algorithm in (*b*). Each element of the heatmaps' diagonals is composed of the sum of such distances of key–presses of the same class, while the rest of the elements result from sums of distances of key–presses of different classes.

The key–press data set is represented in two different ways: using the raw signal data and the *MSD* algorithm. Heatmaps, shown in fig. 4.7, were created over the two 36 × 36 matrices resulting from each of these methods to visualize the distances of key–press comparisons from all classes. The first heatmap, in fig. 4.7(a) is the result of the raw signal representation, while fig. 4.7(b) depicts the results of applying the *MSD* algorithm to the raw signal data to compute key–point descriptors. As

can be seen from both heatmaps, the diagonal, which reflects distances of key–presses of the same class, is composed of lower–valued results compared to distances of key–presses from different classes. Intuitively, this demonstrates that both representations show invariances in their signal description. There is less similarity among different–class key–presses and more for key–presses belonging to the same class. The *MSD* representation visibly improves separability between classes.

4.7.1 Statistical Measures on Class Distances

For each method, I compute some statistical measures on the resulting distance matrix to gain more insight into what the data means. Two groups are investigated:

1. The distances between same–class key–presses, encoded in the heatmap diagonal.

2. The distances between key–presses of different classes, encoded in the rest of the heatmap.

The *f–statistic* and its associated *p–value*, p_f, resulting from a *one–way ANOVA* were recorded, as well as the *z–score*, and its associated *p–value*, p_z. These inferential statistical tests measure separability of groups. The lower each *p–value* is, the more significant are the group differences considered. A $p\text{–}value \leq 0.05$ is generally accepted to mean that the difference between the groups in the data is statistically significant.

Table 4.2: Statistical significance of heatmap results.

	$f\text{–}stat$	p_f	$z\text{–}score$	p_z
Raw Data	58.26	0.00	–7.63	0.00
MSD Representation	2279.20	0.00	–47.74	0.00

The first experimental question paraphrased was: *Does the data obtained from* differential capacitive sensing *indicate some separability between location of key–presses?* Results in table 4.2 show that both p_f and p_z are equal to 0.00, which means that there is an estimated 0% probability of the differences between the two groups having occurred by chance. Inspecting the heatmap in fig. 4.7(a), we can see that the values along the diagonal are low compared to the rest of its values. These observations suggest that the construction of this sensing strategy is viable for touch location identification. However, we can see from fig. 4.7(a) that the raw signal representation, in addition

to having low values for distances between same–class key–presses, also reports low values for other nearby locations. Such results seem reasonable when we consider the design of the sensor relying on one conductive yarn, where *buttons*, or defined key–press locations, are determined along its length. *Buttons* with label numbers that are numerically–close, are located one after the other along the yarn, explaining the low distances among those key–presses as seen in the heatmap. These findings are also compatible with those from [133], where general areas of touch were able to be localized, but not precise locations of key–presses.

In order to determine greater precision in touch location identification, it seems more complex and refined signal representation strategies are necessary. The second experimental question was: *Does the representation produced by MSD increase intra–class similarity and decrease inter–class similarity, compared to the raw signal data?* Results in table 4.2, again, show that both p_f and p_z are equal to 0.00, indicating high statistical significance of differences between same–class key–presses and key–presses of different classes. Furthermore, the reported *f–stat* and *z–score* values for the *MSD* signal representation are considerably larger than those measures for the raw signal data representation, suggesting greater group differences. The corresponding heatmap in fig. 4.7(b) as well, shows how distances between same–class key–presses are much lower than differences between key–presses of different classes. Compared to the heatmap in fig. 4.7(a), the differences between classes that are numerically–close have also been significantly reduced. It can be inferred from these results that the *MSD* representation of the signal does indeed increase the potential for fine location identification. While classification of *button* labels, or key–press locations, is beyond the scope of this study, these results strongly suggest that it is possible to implement touch location classification and construct interactive systems that rely on it.

4.7.2 Resources and Performance

The computation whose speed will be discussed here is that of the heatmap matrix using the data of only one subject, a subset of the whole dataset. It relies on computing the *ELD* of every pair of 180 key–presses, resulting into 32220 total comparisons. The complexity of the *ELD* algorithm, as currently implemented is quadratic. Computing the distance matrix is not a task that would

necessarily need to be repeated for key–press classification, or any application that relies on it. However, since this algorithm has relatively high complexity, it can serve as an indicator of the time each representation of the key–presses might need to be processed depending on the application.

The heatmap matrix for each representation, implemented in Python, was computed on an Ubuntu 18.04 machine with 128 GB of RAM, and an Intel® Xeon® CPU E5–2650 v2 @ 2.60GHz. The CPU has 32 cores, with 2 threads/core, and each heatmap was computed in parallel using 64 processes. The time necessary to compute the heatmap in fig. 4.7(a), where key–presses were represented by raw signal data, was approximately 17 hours, while the time to compute the heatmap in fig. 4.7(b), where key–press representation were produced using the *MSD* algorithm, was approximately 20 minutes. The considerable improvement in processing time is a direct result of the sparse representation of the signal through *MSD*, which is another benefit of using it, in addition to better class separability.

4.8 Conclusion

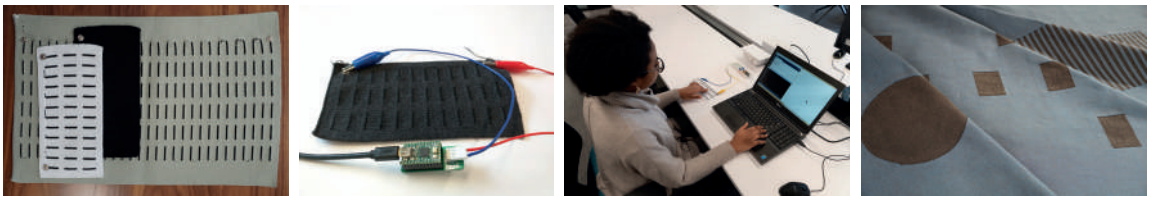
The work presented in this chapter advances the technical capabilities of knitted capacitive touch sensors through the introduction of two novel algorithms, and the application of Bode analysis to existing signal acquisition techniques. Differential capacitive sensing used with Bode analysis increases separability of touch location along a conductive yarn and reduces noise. *MSD*, the feature representation algorithm I introduce, further moves toward accurate sensing by capturing invariant aspects of the signal behaviour that indicate position of touch. Experiments show that the distances between key–presses of the same position are lower than distances between key–presses of different positions. Moreover, the signal representation produced by the *MSD* algorithm outperforms the baseline raw signal data, processed only through Bode analysis and differential capacitive sensing. In order compare the similarity among different key–presses, *ELD*, a distance metric applicable to multi–dimensional time–series data, was used. Supervised learning with respect to positions of touch along the sensing yarn is the next step toward building a variety of accurate interactive touch applications, and will be explored in the next chapter.

Chapter 5: Exploring Real-World Sensing

5.1 Introduction

This chapter introduces techniques used to create interactive systems based on knitted sensors by incorporating machine learning models. Early work showed, through an initial proof–of–concept system, that coarse locations could be differentiated on minimalistic knitted sensors, but highlighted the challenge of precise touch localization [133]. Chapter 4 of this work introduced techniques for high–fidelity and invariant signal representation with respect to touch location, but location identification was not performed, and signal processing was performed offline. Those algorithms serve to characterize the signal obtained from human interaction with minimalistic knitted sensors, however stochastic models are necessary to capture the variability introduced by real–world processes and interactions. Moreover, in order for a user interface to be natural and usable, the requirement of user–specific calibration should be avoided, especially in cases when the same sensor is expected to be used by multiple people.

The work introduced in this chapter enables touch location recognition on the fabric–based capacitive touch sensor [133] for which signal representation was explored in chapter 4. A classification model that accurately localizes touch along the yarn is presented, together with a real–time sensing component, which create the foundation for a future real–time system. This work brings us closer



(a) (b) (c) (d)

Figure 5.1: Knitted sensor designs and applications [3]. *(a)*: Rectangular touchpads of varying size, knitted with carbon–coated nylon and polyester yarn. *(b)*: The real–time sensing controller connected to a knitted touchpad through electrodes at the yarn endpoints. *(c)*: Data collection performed with the 36–button knitted touchpad. *(d)*: Three knitted fabric prototypes of varying patterns composed of the same carbon–coated nylon yarn, and polyester yarn.

to real–world, real–time, and scalable textile sensing by making the following contributions:

- A *recognition model* to identify precise location of touch on an existing knitted sensor circuit relying on one conductive yarn, without subject–specific calibration. This model uses a statistical, frequency–based feature construction component and an LSTM neural network classifier, addressing the challenges of minimal information output from the single–yarn system, as well as noise in the signal.
- A *real–time sensing and signal processing circuit* which obtains input from the knitted sensor. It uses Bode analysis to reject the frequencies most prone to electromagnetic interference (EMI).
- Results from a *series of user studies* to move toward real–time, real–world knitted sensors. The studies enable the creation of a robust location sensing model; demonstrate its generalizability with new users; and start to investigate its robustness under real–world conditions.

5.2 System Description

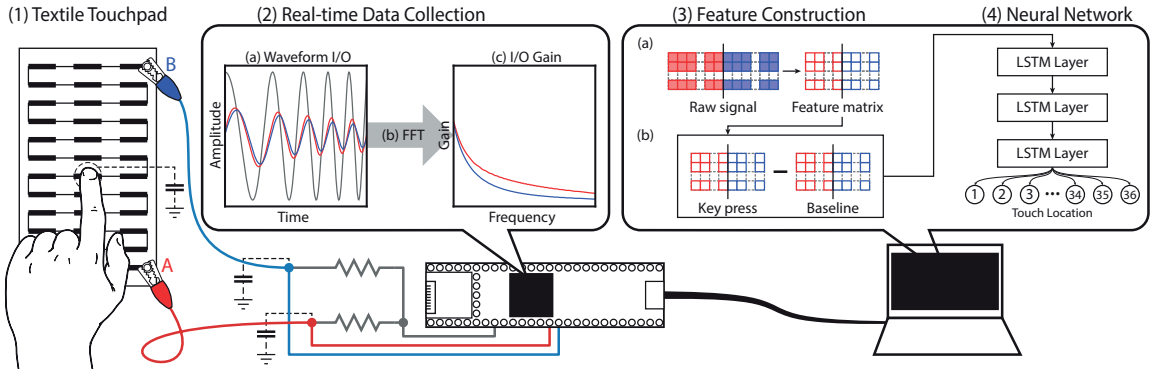


Figure 5.2: Diagram of the end–to–end system components, produced in collaboration with CFF [3]: (1) The knitted touchpad is pressed at one of the pre–defined touch locations; (2) voltage is sourced and measured at endpoints A and B and transformed within the embedded sensing microprocessor to yield the output–to–input gain ratio per frequency; (3) features are constructed based on statistics of the frequency response of measured touch data; (4) the feature representation is used as an input to a neural network to predict the discretized touch location.

To ensure the usability of knitted sensors in the real world, a recognition model enabling precise, real–time touch localization is necessary. The system architecture relies on two main components: the *recognition model* to determine the position of touch on the knitted sensor, and the *real–time*

signal acquisition and processing system from which the recognition system receives data. The model, once trained for high–accuracy recognition, can, in the future, be embedded into a dedicated microprocessor or handheld device, and connect to the signal acquisition and processing controller to identify location of touch in real time. Figure 5.2 provides an overview of the components involved in this process.

The knitted component used with the real–time system and recognition model is the same sensor introduced previously (fig. 5.1(b)), which consists of one conductive yarn as a sensing element and 36 touch locations. As discussed, all locations across a 2–D space are defined along a continuous 1–D yarn pathway, and only voltage readings at each endpoint of the yarn are used as information from the system. Additionally, the signal data obtained is subject to considerable noise. These properties make it relatively difficult to accurately resolve location of touch, compared to other possible designs. It is, therefore, necessary to address the effects of distortion through signal processing and machine learning to ease the constraints placed on the physical textile design and allow for a larger variety of design forms. In addition to these aspects that make high–accuracy location identification a challenging task, preferably, the sensor would be used without user-specific calibration, for a more intuitive and user–friendly interaction.

The results from the previous chapter demonstrated the technical feasibility of constructing an unsupervised system that can differentiate between touch locations on the knitted sensors with one sensing yarn. That work is extended by constructing a real–time signal processing component, using a swept–frequency excitation signal to perform Bode analysis, instead of a single–frequency signal, and adding a learned model to identify touch locations. First, features are constructed over the signal, and then the model is trained using a multi–layer Long Short–Term Memory (LSTM) network architecture, designed to capture the temporal dependencies within the input signals and classify touch location. This section describes the components of the real–time signal processing system, and those of the recognition model for location identification.

The recognition model aims to distinguish between touching each of the 36 sensing locations, and it is trained specifically for this sensor design. For initial evaluation, contact is limited to

single–touch. Other sensor forms could be easily swapped with the existing one, however, the neural network would need to be trained for each sensor form individually, since the interactive components (the carbon fiber touch areas) of the sensor would change depending on the design and needs of the application.

5.2.1 Embedded Sensing Microprocessor

The sensing hardware uses an NXP Kinetis® MK66 ARM® Cortex®-M4F 180 MHz microprocessor (PJRC Teensy 3.6 development board), shown in the second part of fig. 5.2. Two connections lead the fabric circuit to the sensing hardware, which can connect to a computer via USB and transfer processed data at approximately 4.3 Mbaud, and 125 Hz to a *Processing* [110] application for data visualization.

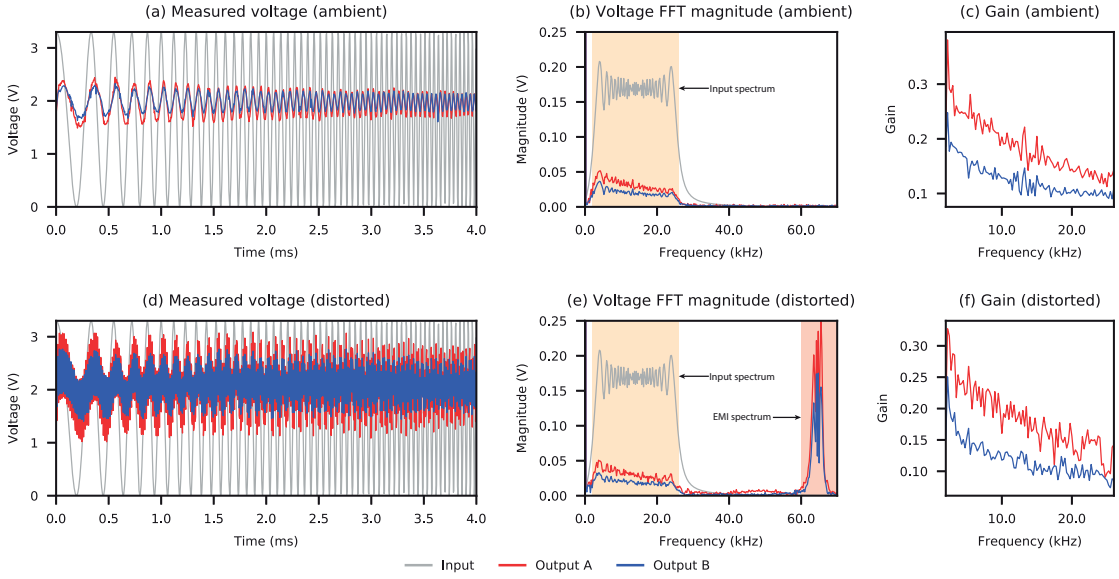


Figure 5.3: Data processing within the microprocessor. Figure produced at CFF [3]. *(a)* A swept–frequency cosine wave is passed through current–limiting resistors and input at both endpoints of the sensing pathway (red). The voltage outputs (green, blue) are measured at the endpoints. *(b)* The magnitude of all three waveforms is computed via FFT. *(c)* The output gain is computed as a function of frequency.

This system analysis considers the contributions of attenuation across multiple frequencies through Bode analysis, while the analysis described in the previous chapter relied on a single frequency attenuation. This strategy provides a greater amount of information to describe touch, and accounts for variance in any single frequency bin through consensus. In order to control the frequency com-

position of the input waveform, a linear swept–frequency cosine wave is generated, which is stored in memory and then output by the microprocessor's internal 12–bit, 0 V to 3.3 V digital–to–analog converter (DAC).

Figure 5.3 illustrates data flow within the microprocessor. Two 16–bit analog–to–digital converters (ADCs) synchronized with the DAC sample the output waveform at approximately 256k samples–per–second in 1024–point increments. Fast–Fourier Transforms of the samples are processed using the ARM CMSIS DSP library complex FFT function [72]. Ninety six frequencies between $f_0 = 2,000$ Hz and $f_1 = 26,000$ Hz were input for a duration of $t_0 = 0$ seconds to $t_1 = 0.004$ seconds. The waveform uses the full range of the ADC's voltage output, between 0 V and 3.3 VDC, thus the amplitude, A, and bias, b, are 1.65 V and 1.65 V respectively. To characterize the system response during data analysis, the gain values of all recorded frequencies are used, for a total of 192 values per record window.

The swept–frequency range is chosen based on the locations of frequencies affected by additive distortion (fig. 5.3e). The two significant sources of distortion are *power supply ripple*, contributed by the microprocessor's DC power source, and *electromagnetic interference* (EMI), induced by external power sources in proximity. Power supplies that transform AC to DC voltage often induce a small amount of ripple into the rectified output. Ripple is a small fluctuation in the device's operating voltage, measured at approximately 200 mV at 60 Hz in ambient conditions. Background on these phenomena is provided in section 2.3.2.

Human touch contributes to measured distortion, increasing the ripple amplitude to approximately 1 V. Because capacitive sensors rely on a voltage difference to charge and discharge a capacitor in contact with the sensor, and induce a controlled oscillation, this phenomenon contributes to the induced oscillation to create unexpected voltage shifts in the time domain. Electromagnetic interference is radiated distortion produced by circuits transmitting power. The severity to which the EMI affects surrounding electronic devices depends on the quality of EMI shielding. Capacitive sensing relies on exposed, unshielded wiring to measure touch, and thus is more susceptible to EMI [119]. The distortion shown in fig. 5.3d and fig. 5.3e is generated by a 70 Watt fluorescent

lamp in close proximity, approximately 18 in., to the fabric sensor. An interference band of 10 kHz centered at 65 kHz is observed as a resultant of the EMI. The proximity of the lamp affects the induced distortion.

Once the signal is acquired and Bode analysis is performed, in an interactive system it would need to be sent to a model capable of interpreting it as a location of touch. The subsections below describe the *recognition model*, composed of a feature construction step and the neural network architecture. Its purpose is to translate the acquired signal data into one of the 36 discrete interactive components of the knitted touchpad.

5.2.2 Feature Construction

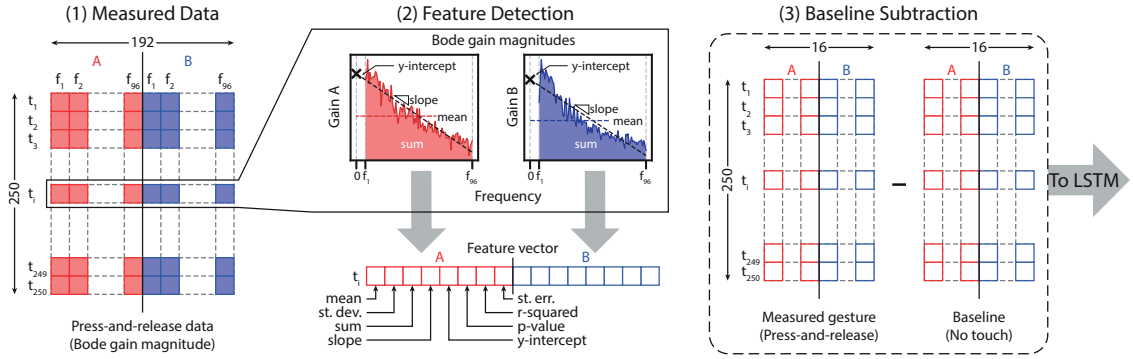


Figure 5.4: Feature construction over frequency gain values of both signals. Figure produced in collaboration with CFF [3]: (1) the raw data is composed of 250 time instances and 96 frequency values per signal, for a total of 192; (2) statistical features are computed over the frequencies per time step; (3) baseline is subtracted from key–press data.

The system constructs features from the input–to–output voltage gain values of the sensor over time. A touch event is measured over two seconds. This allows enough time to capture the event onset, saturation, and offset. Each touch event consists of 250 temporally–related measurements of the voltage gain values, which are captured as explained in section 5.2.1. This amount of time is more than enough for the voltage discharge which defines the touch event to occur. Since there are 96 frequency responses recorded from each of the two sensor endpoint signals, one touch sample is represented as a 250 $\times$ 192 matrix before feature construction, associated with its respective touch location or *button* number as a label. Statistical features are constructed over all the captured frequency responses for each signal for each of the 250 measurements in the sample, and the results

are concatenated to form the features. The *mean, standard deviation*, and *sum* are calculated over all gain values for each signal and each time step. Additionally, a line is fitted to those frequency response values and the *slope, intercept, p–value, r–squared*, and *standard error* are included as features for each signal, representing each touch sample as a 250×16 matrix. The *standard error* is the average distance of all samples from the fitted regression line. *R–squared* is the percentage of the dependant variable variation that is explained by the linear model. The *p–value* is a measure of how changes in the independent variable affect the dependent variable, with a low *p–value* indicating that the independent variable is a meaningful addition to the linear regression model. This way, for every time step, the system is represented in terms of its frequency gain statistics to increase stability compared to the raw frequency gains.

Ultimately, each touch event is characterized as the difference between the captured touch sample represented in terms of the statistical information of its frequency gains, and the same statistical features computed over a baseline measurement where no touch occurs. The baseline is meant to capture the conditions of the surroundings, and remove their representation from the touch event characterization, so that the information that remains is more indicative of the properties of touch position. Figure 5.4 visualizes the feature extraction process.

5.2.3 Neural Network Model

Data processed as described above is used as an input to a neural network, which relies on three recurrent layers, followed by one fully–connected layer of 100 nodes, and an output layer of 36 nodes, as illustrated in fig. 5.5. The network's three recurrent layers are based on LSTM cells. LSTMs have been widely used for sequential data analysis, including time–series signal data, since they capture temporal dependencies. Section 3.2.2 provides an overview of LSTMs and their benefits.

The sequence of the frequency gain statistics captured from each electrode signal is used to construct the input to the network. Each input consists of one key–press on the touch location, represented as a sequence of 8 different statistical measures computed over values of all frequencies from each of the two electrodes. The captured sequence is 250 time steps long, with 16 values per time step. Therefore, the input layer of the neural network architecture has a size of 16. The frequency

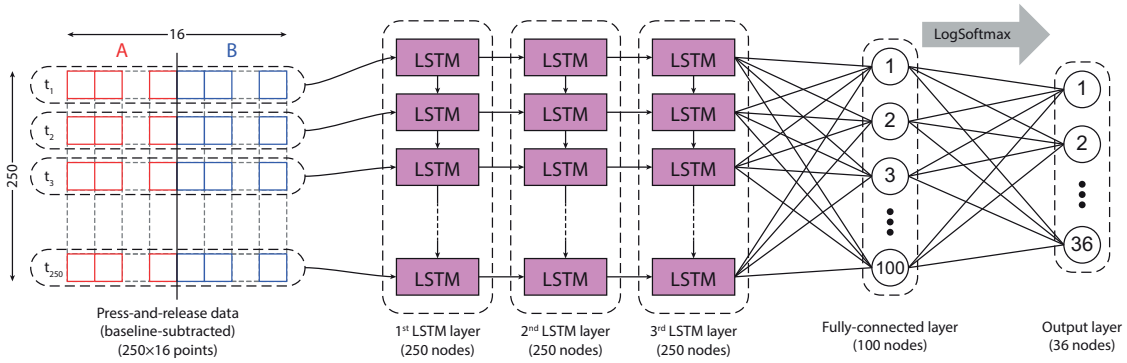


Figure 5.5: LSTM Neural Network model. Figure produced in collaboration with CFF [3]. The network is composed of 3 subsequent layers of LSTM capturing the temporal dependencies. The last layer is linear, which then translates the input to one of the 36 button positions, through a LogSoftmax activation function.

gain statistics are temporally–related, which the neural network architecture aims to capture. The size of the output layer is 36: the final linear layer maps each input into a *button* position on the knitted touchpad through a LogSoftmax activation function. The above description corresponds to the architecture of the network, however, to create an accurate model, this network needs to be trained with user data, so that its weights are adjusted accordingly. Section 5.3 below describes the training and evaluation process for the recognition model.

5.3 Model Training and Evaluation

To move knitted sensors toward real–world use, three studies of the system described above were conducted. The first study consisted of collecting user data to create a model of interaction that would allow accurate location sensing in real–time, provided the hardware infrastructure is in place for real-time sensing. Then, to test the accuracy and robustness of the system, two more user studies were conducted: one validation study to evaluate the performance of the model on new data, and another to test its robustness to everyday use activities. The robustness study consisted of two conditions: *(1)* proximity to a fluorescent light lamp, as a source of EMI, which as discussed above, causes a disturbance in the signal of capacitive circuits, and *(2)* stretching of the sensor, which alters its geometry. I evaluated the system's performance under each of these conditions on the trained model resulting from the first user study. The accuracy results reported from all three studies are

subject–independent: no calibration was performed for individual subjects. I discuss these results in a usability context, as well as compare the proposed recognition model to other benchmark algorithmic solutions.

5.3.1 Study 1: Model Construction and Cross–Validation

This study was performed using the knitted touchpad illustrated in fig. 5.2. I trained the neural network described in section 5.2.3 to be able to classify the button that was pressed. The model created during this study can be used in conjunction with the real–time signal acquisition and processing system (section 5.2.1) in order to classify location of touch.

Dataset Description:

In order to create a trained neural network model, button–press data from 24 subjects, who were college students, was collected. Each subject conducted 10–20 trials, with one trial consisting of pressing each of the 36 touch locations on the knitted pad once. There was a total of $13,896$ labelled key–press data instances: 386 samples of key–presses for each of the 36 touch locations on the keypad. Each key–press sample had a size of 250×16.

Methods and Resources:

Data was collected and saved to a computer running the *Processing* IDE. A Processing sketch was used to visualize and save the frequency gain values from each electrode, which were returned by the Bode analysis in real–time. A Keysight MSOX3024T Oscilloscope was used to visualize the input and output waveforms in real–time. A more comprehensive data acquisition and signal processing description can be found in section 5.2.1.

First, features were constructed from these raw data, as described in section 5.2.2. The resulting representation was then used to train a neural network and tune its parameters through 10–fold cross–validation. On every iteration, all the data from 2 or 3 subjects was used for validation, while data from the other subjects was used to train the model. The dataset was balanced for all classes, since subjects were required to press on every button for each trial. The network weights were initialized using Xavier initialization [43].

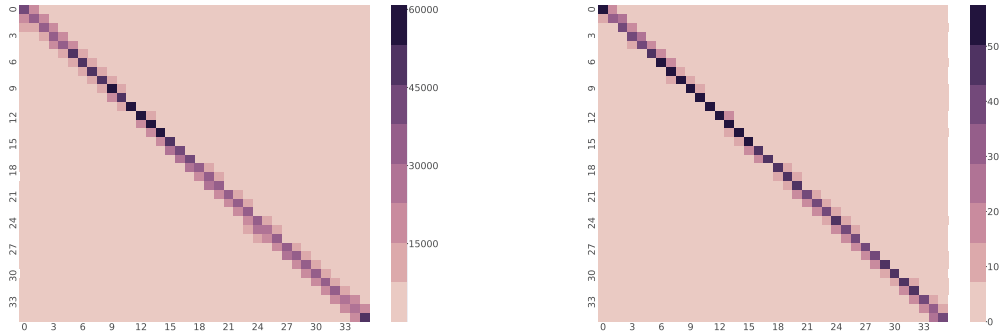
The model was trained using PyTorch 1.4 [104] with CUDA 10.0, using a learning rate $\alpha = 0.001$, a dropout rate = 0.6 and a batch size = 128, over 2000 epochs. The optimizer used was *Adam* and the loss function was negative log likelihood. The machine was running Ubuntu 18.04, with an Intel® Core™ i9–9960X CPU @ 3.10GHz, 128 GB RAM, and 2x RTX 2080 Ti Blowers with NVLink cards.

Results:

To evaluate the trained model, I calculate the accuracy of identifying touch location on the knitted component. *Accuracy* is the ratio of correctly predicted key–presses to the total number of key–presses evaluated. As previously mentioned, there were 36 classes corresponding to 36 *buttons* or touch locations, thus, chance accuracy is 2.78%. This model achieved an accuracy of 58.3%, which was calculated as the average validation accuracy of all 10 folds. For each fold, accuracy was calculated as the average of the achieved validation accuracies of the last 50 epochs. Table 5.1 shows accuracy results of all user studies, and fig. 5.6(a), the average classification matrix of this study, which was computed over the classification matrixes of each of the 10 models of cross–validation. The *macro–precision*, *macro–F1–score*, and *macro–recall* are also calculated. Precision refers to the ratio of true positives to the combined number of true and false positives. Recall is the ratio of true positives to the combined true positives and false negatives. The F1–score refers to the harmonic mean of precision and recall: $F1 = 2 * (precision * recall)/(precision + recall)$. Macro–precision, macro–recall, and macro–F1 refer to the balanced respective scores per class, and those values are obtained by averaging all respective class scores.

5.3.2 Study 2: Evaluation with New Sensor Touch Data

In order to determine the accuracy of the neural network model, sensor touch data was collected from 7 other users, whose data had not been previously seen by the trained model, either in the training or validation set. The purpose of this study was to evaluate the performance of the system under normal conditions—similarly to the ones when training was conducted. Even though cross–validation was performed above, several network architectures and parameters were explored to



(a) (b)

Figure 5.6: Classification matrices produced with results from study 1 *(a)*, and study 2 *(b)*. The matrix rows denote the true row categories of the key–presses, while the columns show the ones predicted during evaluation. Each category is denoted by the respective button number of the knitted touchpad. A higher–value diagonal, with the rest of the matrix having lower values, would be a desirable combination indicating a high accuracy. This figure illustrates the challenges of classifying arbitrarily defined positions on a continuous element—the buttons are sometimes classified as the ones adjacent to them.

increase validation accuracy, which might have introduced bias in this model. A new dataset tested against the trained model serves to obtain a more objective evaluation of its performance.

Dataset Description:

The subjects of this study were 7 college students, and the experiment set–up was the same as during the initial study, described in section 5.3.1. Subjects pressed each of the 36 keys of the rectangular knitted sensor keypad 10 times for this evaluation, for a total of 2,520 key–presses, or 70 per button—the sensor defined area of touch classification. Both the true labels, which were the button positions on the sensor, and the labels predicted by the model were recorded.

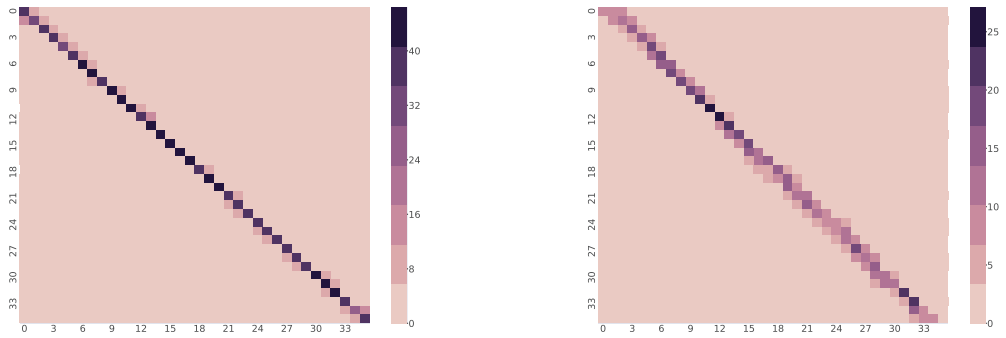
Methods and Resources:

The collected data was used as a testing set towards the neural network models created in section 5.3.1. Features were constructed on the key–press signals before they were used as inputs to the trained model for evaluation. Testing was performed on the same machine as model training, described in section 5.3.1.

Results:

An average accuracy of 66% was achieved during this experiment, as recorded in table 5.1. This accuracy was calculated by testing the new data against each of the 10 training models created during cross–validation, and subsequently taking the average of the results. Figure 5.6(b) shows the corresponding classification matrix. Similarly, this classification matrix was generated as the mean of all classification matrices produced from the data tested against each trained cross–validation model. Its respective macro–precision, macro–recall, and macro–F1 scores are also reported in table 5.1. These results, which are even better than the original validation results from study 1, indicate the robustness of the proposed model.

5.3.3 Study 3: Robustness to Sensor Distortion



(a) (b)

Figure 5.7: Classification matrices representing the accuracy of experiments in study 3 condition 1 *(a)*, and study 3 condition 2 *(b)*. The matrix rows denote the true row categories of the key–presses, while the columns show the ones predicted during evaluation. Each category is denoted by the respective button number of the knitted touchpad. Diagonal elements having relatively high values, and non–diagonal matrix elements having lower values would be a desirable combination, indicating high accuracy. Similarly to fig. 5.6, sometimes buttons are mistakenly classified as the ones close to them, due to the continuity of the sensing yarn.

These sensors are intended to work in everyday life, behaving like fabric, and resisting signal distortions. For this reason, the fabric sample on which the recognition model was trained was exposed to two different types of sources that could cause signal distortion: *(1)* electro–magnetic interference and *(2)* stretching.

Dataset Description:

Touch data was recorded from 4 different subjects for *condition (1)*, which consisted of exposing the sensor to EMI through a fluorescent light bulb, placed 12–18 inches away from the sensor. Additionally, data from 3 other subjects was recorded for *condition (2)*, with the sensor being stretched. Each subject pressed the 36 buttons of the sensor 10 times. Similarly to the other two data–collection user studies, subjects were college students. For *condition (1)*, the dataset consisted of $1,440$ key–press samples; for *condition (2)*, there were $1,080$ key–press samples.

Methods and Resources:

The methods used in this study were the same as those of study 2. For each condition, its respective dataset, after being processed through feature construction, was used to evaluate the model created in study 1. The same machine was used to run these experiments, as that in studies 1 and 2.

Results:

Table 5.1: Classification results for the three studies. Reported measures include: the average accuracy of identifying the location of touch on the knitted touchpad across the three user studies; the balanced F1–score, which is the harmonic mean of balanced precision and balanced recall; the balanced precision; the balanced recall.

Study	*Accuracy*	*Macro–F1*	*Macro–Precision*	*Macro–Recall*
Study 1	0.583	0.516	0.516	0.518
Study 2	0.662	0.662	0.665	0.662
Study 3–1	0.795	0.794	0.796	0.795
Study 3–2	0.372	0.364	0.393	0.372

The results of these experiments, summarized in table 5.1, and fig. 5.7, were computed in the same way as those of study 2. An accuracy of 80% was reached for *condition 1*, while the accuracy for *condition 2* was 37%. From these results, we can see that the model retained its robustness even under conditions of signal and sensor distortion, especially during *condition (1)*, which is promising toward using it in the real–world. In *condition (2)* we see that the model does not hold as well against the structural distortion of stretching.

5.3.4 Benchmark Comparisons to Existing Techniques

To demonstrate the utility of the proposed recognition model for robust location sensing, I compare its performance to three other related benchmarks. In particular, the reported results include: using a single frequency Bode analysis (*benchmark 1*); using the MSD algorithm for feature construction (*benchmark 2*); using a multilayer perceptron (MLP) neural network classifier (*benchmark 3*). These three benchmarks are described below, and their results summarized in table 5.2.

Table 5.2: Accuracy results of our recognition model as well as other 3 other benchmark models.

Method	*Frequency*	*Features*	*Classifier*	*Study 1*	*Study 2*	*Study 3–1*	*Study 3–2*
Benchmark 1	Single	Raw	LSTM	0.253	0.256	0.313	0.203
Benchmark 2	Single	MSD	LSTM	0.050	0.057	0.051	0.081
Benchmark 3	Multi	section 5.2.2	MLP	0.463	0.435	0.560	0.258
Our model	Multi	section 5.2.2	LSTM	**0.583**	**0.662**	**0.795**	**0.372**

In *benchmark 1* the same LSTM classifier is used, however the features are different. The purpose of this comparison is to investigate the effectiveness of using the feature extraction process described in section 5.2.2, in conjunction with multiple frequency gains obtained from the Bode analysis. Only one frequency is input into the system, in contrast to 92 different frequencies used in our introduced method. The frequency gains are measured at each of the two sensor endpoints, as described in section 4.2. For this benchmark method, no feature construction was used: the features are composed of the raw frequency values of the two endpoints, therefore each key–press in this case is represented as a $250x2$ matrix. The neural network and its settings were the same as the ones used in the proposed recognition model, described in section 5.2.3 and whose results are summarized in table 5.1. The work presented in chapter 4 however, did not include classification. The LSTM step was added in the benchmark method for consistency, in order to make the related aspects of the two projects comparable.

Benchmark 2 uses the same frequency value as *benchmark 1* and an LSTM neural network, but it applies the MSD algorithm, introduced in section 4.4, as a feature construction step. We can see from these results that the current recognition model would outperform the work in chapter 4, even if that work included the same classification step, assuming MSD was applied in this way to

the singe–frequency data. MSD could potentially be applied to the signal in different ways for high accuracy, such as a feature construction step to the raw signal data of all 192 frequencies. It can also be combined with other feature extraction methods in more complex ways, including the one presented here. Alternatively, it can be integrated into a custom neural network. However, these explorations are beyond the scope of this work.

Benchmark 3 consists of a model which has the exact feature construction steps as the one introduced in section 5.2.2, including multiple frequencies. The difference is in the neural network classifier. Instead of an LSTM, it uses a Multi–Layer Perceptron (MLP) network, which does not necessarily retain time dependency information. All the 250 processed time instances of the key–press are concatenated and input into the network, labelled with the button position. As can be seen from table 5.2, the proposed recognition model outperforms the benchmarks in all experiments. Below, these results are discussed more in depth.

5.3.5 Results Summary

The recognition model was trained through the cross–validation (study 1) for accurate touch detection on knitted sensors, and the evaluation study (study 2) confirmed the validity of these results. However, such sensors are not intended to be used in controlled lab environments, and so the model performance was explored under conditions that would cause signal distortion (study 3).

The results of the first three studies are summarized in table 5.1, which includes average accuracy, macro–F1 scores, as well as precision and recall. These measurement values are close to each–other, showing balance in the constructed model. Additionally, as can be seen from fig. 5.6 and fig. 5.7, most cases of buttons being classified incorrectly were the ones classified as buttons positioned next to the ones pressed.

Study 3 shows that the model is mostly consistent, even when exposed to some distortion. Study 2, which included testing new data under regular conditions, shows an increased accuracy compared to the cross–validation results, which indicates model stability. In study 3, *condition 1* of the experiment reached even higher accuracy results. This indicates no visible or measurable negative effect of exposing the sensor to EMI radiation on its ability to localize touch, which is a highly

desirable quality for a sensor. In study 3, *condition 2* showed that stretching the sensor decreases its ability to accurately identify location of touch, primarily because the resistance of the individual buttons changes. In the future, this aspect can be addressed by designing algorithms with more robust spatial resolution. However, these results are still encouraging, since they mean that this sensor can behave like fabric and retain some touch recognition accuracy.

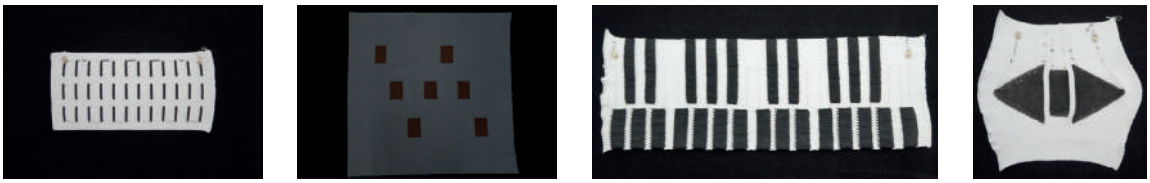
It is worth noting that this model was not trained with data collected under the two distortion conditions with which it was experimented (EMI and stretching). The expectation is for the accuracy of these tests to have been higher, had this network been trained on such data. The main reason for this choice was to investigate the effect of unknown possible distortions. For a finished product implementation, training should be performed under a much larger variety of conditions, including the ones mentioned above. Moreover, the accuracy results introduced here are subject–independent to make the experience of multiple users interacting with the same sensor as intuitive and non–intrusive as possible. However, if a knitted pattern is designed to be used by a single user, calibration is likely to increase the location identification accuracy, since it removes variations that are caused by different users' physiological states.

Finally, the benchmark studies put the recognition model results in the context of other proposed methods, to demonstrate its effectiveness and reliability. In the next section, several existing knitted design forms using the same construction method of one yarn and two connections are discussed in terms of touch location identification. Any of them would be able to be directly plugged into the existing system to output location of touch, assuming model training with user data training for each sensor geometry.

5.4 Design Patterns and Resistance

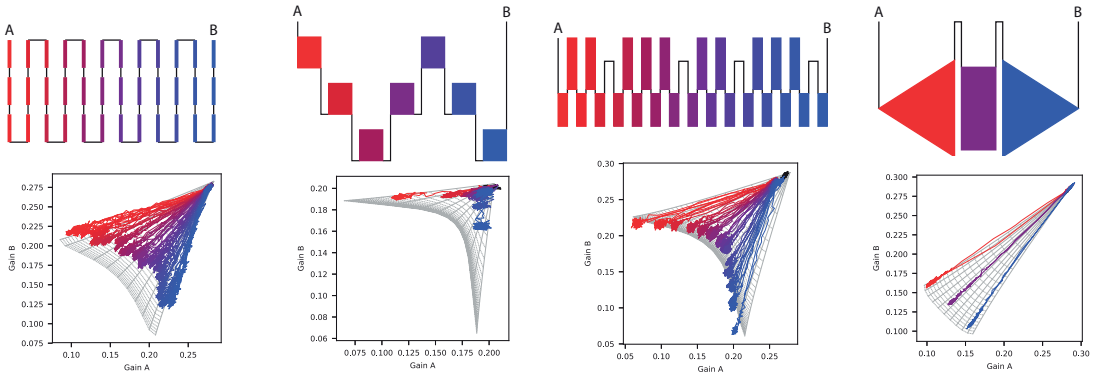
Section 2.1.2 of this work describes how different sensing patterns can be designed using the same process as the sensor used in this work so far. Different geometries would translate to different electronic properties of the circuit, impacting location identification accuracy.

While designing these sensor geometries, it is important to abide by the knitting process specifications and requirements, since the serpentine routing of the yarn is a function of the knitting



(a) (b) (c) (d)

Figure 5.8: Knitted sensors of different design patterns described in section 2.1.2 [3]: *(a)*: the 36–button touchpad, which was used in the system pipeline and user study evaluation above. *(b)*: the multi–button game controller *(c)*: the 25–button piano keyboard *(d)*: the media control button pad



(a) (b) (c) (d)

Figure 5.9: Comparison of gain measurements towards touch separation for button sensors. Figures produced at CFF [3]. The top right corner of each plot denotes the baseline, no–touch state. The rest of the plot shows the signal response to touch on every button location. *(a)*: the 36–button touchpad sensing area *(b)*: the multi–button game controller sensing area. *(c)*: the 25–button piano keyboard sensing area. *(d)*: the media control buttons sensing area.

machine. When designing with the purpose of connecting a knitted piece to a circuit, patterns that would cause shorting of the circuit would need to be avoided. Such patterns involve crossing of conductive elements (e.g. a T–shape). Additionally, it is important to consider the length and resistance of the yarn pathway between contact points. Resistance of a conductor depends on its resistivity, cross–sectional area, and length. Resistivity and area of the conductor do not change throughout each sensor, since the conductive yarn is homogeneous. Resistance increases proportionally with the length of the yarn. This means that in areas where the yarn is knitted more tightly together, the resistance is higher. A large resistance limits current flow and increases the capacitors' charge time compared to a more conductive pathway. This improves measurement sensitivity given the same magnitude of induced capacitance.

Figure 5.8 shows four different knitted touchpads and fig. 5.9 compares their geometries by illustrating their respective touch localization performances. Each touch position is plotted in relation to two values: the gain to the single–frequency input response from each sensing endpoint captured through previously–introduced methods [133], without any algorithmic or machine learning approaches introduced in this work applied. The data was collected from a single subject pressing on each defined sensing area one time, and is provided for illustration purposes. It does not constitute a complete evaluation of these sensors. Some technical and construction details of the above–mentioned design samples are provided in table 5.3. The cross–touchpoint resistance refers to the resistance measured between the start and the end of one defined touch location, such as a button. The inter–touchpoint resistance is measured between the end of one touch location and the beginning of the other. Only these sensors are illustrated because their primary mode of interaction is discrete touch location identification, not pressure sensitivity or continuous swiping input usable for gestures. Those interaction modalities should also be investigated in more depth in the future.

Table 5.3: Detailed technical properties of sensor prototypes.

Design Form	*Size*	*Overall Resistance*	*Cross–TP Res.*	*Inter–TP Res.*
36–Button Touchpad	4×8 in.	550 kΩ	8.5 kΩ	15 kΩ
Multi–Button Game Controller	36×3 in.	1.75 MΩ	3.68 kΩ	213 kΩ
Piano Keyboard	15×4.75 in.	783 kΩ	5.99 kΩ	24.26 kΩ
Media Control Buttons	7.25×6.25 in.	206 kΩ	8.13 kΩ	47.99 kΩ

As can be seen fig. 5.9, the touch location data is easier to separate into clusters even analytically, when the conductive yarn is knitted in larger areas. Figure 5.9(d) is an example of a relatively straightforward separation between touch location. The structure illustrated in fig. 5.9(a) poses a considerably greater challenge to accurate location identification, because the sensing yarn is not assembled closely together, and its inter–button resistance is low (table 5.3). This chapter explored techniques to advance the capabilities of a sensor knitted with this yarn pattern, particularly because it is the most challenging of the patterns introduced here, and it stands to reason that it would be easier to identify location of touch on the other geometries. However, the trained recognition model presented here was sensor–specific, as experiments have not been conducted with other patterns. The

sensor forms presented in this section whose mode of functionality is detecting discrete locations, illustrated in fig. 5.9, can be plugged in to the hardware system introduced instead of the 36–button knitted touchpad. The same feature construction step and neural network architecture can also be used for touch location detection. However, each pattern needs to have its model trained with its own user data, so the model can implicitly learn the position of touch for that specific geometry. Future work should explore more generalized algorithmic and machine learning solutions to enable detecting location of touch on any yarn routing path.

5.5 Discussion and Conclusion

The work introduced in this chapter explored the potential for knitted sensing fabric to be used as a viable embedded interface, enabling interaction. The main focus was building a recognition model to identify fine location of touch on a 36–button sensor. Through three user studies, and comparisons with benchmarks, two key aspects of bringing these sensors to the real world have started being addressed: increased accuracy, and robustness of the system. I constructed a recognition model which uses a LSTM neural network to classify the location of touch on the knitted sensor with 36 touch locations. These touch locations were pre–defined from the sensor design, and a touch event was recorded in real time through a dedicated signal acquisition and processing system. The model was trained using data from 24 subjects, and achieved an average subject–independent accuracy of 58% from the different cross validation folds. Subsequently, the model was evaluated using new user touch data from 7 new users under normal circumstances, achieving an accuracy of 66% on the 36–button sensor. The same previously–trained model was also evaluated under conditions of two possible distortions: the effects of electromagnetic interference (80% accuracy), and while the knitted section was being stretched (37% accuracy). Finally, these results were compared with several related methods as benchmarks (using only one frequency response with Bode analysis; using MSD feature construction; and using a different neural network classifier) to further demonstrate the robustness of the proposed models.

Other sensor designs would need specific training in order to perform that functionality. However, the same real–time sensing system component, feature construction method, and neural network

architecture can be used for different sensor geometries. The technical contributions and studies conducted in this work helped lay the foundation to advancing towards viable and robust interactive knitted textile touch interfaces. Next, considerations to bringing such sensors to the real world are explored more in depth, through a formative study to understand users' views, and some everyday use experiments.

Chapter 6: Interaction with Knitted Sensors

6.1 Introduction

The purpose of the work described in this chapter is to start shaping the future of ubiquitous applications based on minimalistically-designed knitted sensors according to user expectations and real–world conditions. Chapters 4 and 5 discussed steps to make touch localization more accurate and take this technology closer to real–world use. In addition, this sensor construction technique enables the easy production of several different design forms relying on the same design principles of one conductive yarn and two external connections. These design aspects and advancements show considerable potential toward transforming such sensors into truly pervasive technology, enabling a variety of interactions. However, in order to properly explore their application space, the end users' perspective and everyday use scenarios should also be integrated in the process.

While the focus of this investigation on end–users' views is touch–sensitive knitted fabrics constructed using this particular design strategy, many of the findings could be extended to fabric sensors of different design and construction methods. Fabric–based touch sensors, created through various techniques, such as embroidery [107, 42, 52, 53, 9, 94], weaving [29, 55, 127, 7, 108, 143], and knitting [106, 141, 30, 133, 132, 93], have shown great potential towards enabling interactive applications through flexible and durable interfaces.

To start understanding users' perspectives, I conducted a formative study, during which participants were shown several of the design samples, and inquired regarding their views and areas of interest related to this technology. Informed by the focus group study, some technical aspects of the touch–sensitive fabric design were subsequently explored. These are aspects about which users in the formative study also inquired, particularly related to everyday usage scenarios. This investigation fills a gap in existing literature since qualitative user studies from existing work have been narrower in scope, primarily focusing on specific design aspects. Informed by the focus group study, some technical aspects of these particular touch–sensitive fabrics are explored. These explorations provide

(a) (b) (c)

Figure 6.1: Touch-sensitive knitted fabric and example applications [3] used during the formative focus group study. *(a)*: A large multi-button knitted controller used with the *Whack-a-Mole* game. *(b)*: A knitted controller on which the touch-sensitive area is designed to capture sliding motions and gestures. *(c)*: A system of knitted components: a piano keyboard, volume slider, and media control button touchpad connected, via I2C, with a single sensing controller directing communication among all three devices. The master I2C device connects to a MacBook Air via USB MIDI to interface with *GarageBand*.

insights related to designing interactions and implementing applications relying on these sensors for user input. As mentioned, some of the findings of this work can be generalized to the broader field of smart textiles, while other considerations, particularly those related to technical evaluations, are more specific to the construction of sensors relying on this design process and philosophy. The contributions of the work in this chapter are the following:

1. A report on a qualitative formative user study with 32 participants, structured as 8 focus groups is produced. During this study, I introduced users to the touch–sensitive knitted fabric technology and its basic working principles, presented them with several design samples and applications prototypes from prior work, and asked questions regarding their opinions and perceptions.

2. Through thematic analysis of the focus group data, I identify new research directions and create a basis for later building real–world applications relying on touch–sensitive knitted fabrics according to users' views. I also identify specific hesitations and expectations of potential end users from the formative focus group study. An underlying requirement to developing the desired applications was the ability to use gestures in addition to simple touch on the fabric. Furthermore, there were frequently mentioned concerns about safety and everyday use durability.

3. Informed by the focus group study, the capability of one of the fabric designs to differentiate among three different, but related swipe gestures, as well as simulated accidental touch events, is investigated. A user study with 12 users was conducted, where each user performed each gesture several times on the sensor, to later compute the similarity between all captured samples using the *Euclidean Levenshtein Distance (ELD)* metric, introduced in section 4.5.

4. The durability and everyday use potential of the touch–sensitive knitted fabric technology is demonstrated as well. To do this, washing and drying experiments were conducted on three different touch–enabled knitted fabrics, exploring the effect that this process has on the sensor resistance. Resistance is an important property of the conductive area, since it affects the signal output from it. Similarly, the effect of stretching, horizontally and vertically, on the resistance of the sensors was tested. These experiments are necessary for informing future high–fidelity computational models to characterize the behavior of knitted sensors of different designs during everyday use.

6.2 Formative Study: User Insights

In order to investigate the usability potential of touch-sensitive fabric in general, and minimalistically-designed knitted sensors in particular, I conducted a qualitative formative study in the form of focus groups. The purpose of this study was to understand users' perceptions of incorporating interactivity in their environments and clothes, enabled by touch sensitive knitted fabrics. Participants were invited to interact with several textile sensors and applications to explore their potential benefits, as well as envision other possible forms. These sensor examples [133, 3] were introduced in section 2.1.2, together with application prototypes, and their properties related to circuit design were further discussed in section 5.4. Table 6.1 summarizes them. The objectives of this study were the following:

- To understand users' views and experiences regarding interaction with this technology
- To explore application areas into which users would want this technology integrated
- To identify potential concerns when engaging with this technology for further study

- To generate design guidelines for future interactive applications based on touch-sensitive fabrics

(a) (b) (c)

Figure 6.2: Touch sensitive fabrics of varying sizes and designs, knitted with conductive carbon-coated nylon and regular polyester yarns of different colors [3]. They are all knitted in one piece with no electronic layers in between. *(a)*: Front of touchpads of varying sizes and colors. The noticeability of interactive components can vary with the non-conductive yarn choice: the black touchpad has subtle interactive areas, while the white one prominent ones. *(b)*: Back of touchpads of varying sizes. The bigger sensor is knitted to be thicker through the same process. *(c):* Multi-button controller sensors with a similar design but different sizes. The knitted components on each sensor are scaled versions of each-other.

6.2.1 Participants

The study was structured as 8 focus groups of 2-5 participants each, with a total of 32 participants. Participants were undergraduate and graduate students of a median age of 24 years old, with 20 of them being male, 11 female, and 1 of non-binary gender. Each participant signed an informed consent form approved by our institutional review board before entering the study.

6.2.2 Procedure

During each focus group discussion, I moderated the discussion, asking a series of pre-defined questions and other follow-up ones as necessary, to ensure that the users' expressed ideas were understood. An observer recorded the conversation and noted important ideas, themes, quotes, impressions, and body language [76]. Audio recordings were also taken for reference. Participants were first given a brief introduction to the touch-sensitive knitted fabric that is the focus of the study. In addition, the design philosophy behind it and its basic working principles were explained. The discussion featured three main phases, described below.

Table 6.1: Characteristics of design form factor used in the formative focus group study.

Sensor	*Size (in)*	*Figures*	*Description*
Small Black Touchpad	8x4	6.2(a), 6.2(b)	A knitted sensor constructed by routing a conductive yarn through the fabric during the knitting process to create 36 interactive, button-like points across its surface [133, 132, 93]. The black carbon-coated yarn is less visible combined with the black polyester.
Small White Touchpad	8x4	6.2(a), 6.2(b)	A knitted sensor constructed by routing a conductive yarn through the fabric during the knitting process to create 36 interactive, button-like points across its surface [133, 93]. The black carbon-coated yarn is more prominent against the white polyester.
Large Touchpad	18x10	6.2(a), 6.2(b)	A knitted sensor constructed by routing a conductive yarn through the fabric during the knitting process to create 180 interactive, button-like points across its surface [133, 132, 93].
Keyboard	12x3	6.3(b), 6.3(d)	The touch points of this sensor are knitted in the same serpentine structure as those in the touchpads, though the placement is staggered to mimic a alphanumeric QWERTY keyboard [132].
Single Button Controller	36x36	6.7(a)	This prototype has a large circular sensing area in the middle. It can be used as a controller for applications that rely on pressing a button, as well as controlling pressure [93].
Multi-Button Controller	36x3	6.1(a), 6.2(c)	This design is composed of several square-shape sensing areas, with the primary interaction modality being pressing one of the large sensing locations [93].
Slider Controller	36x3	6.1(b)	This 32-row slider pattern is intended to capture continuous input, such as *swiping motion* [93].
Volume Slider	6x7	6.1(c), 6.4	A 22-row slider, which was knitted to control the volume in the *GarageBand* application. It relies on a swiping gesture used to control continuous input, similarly to the *Slider Game Controller* described above, but its proportions are smaller [93].
Piano Keyboard	15x5	6.1(c)	This prototype is a 25-button touch-pad, knitted to emulate a 2-octave piano instrument. The design and spacing of its keys follow those of a regular piano keyboard, and it can be used as a controller for the *GarageBand* application [93].
Media Control Buttons	7x6	6.1(c)	The button pad contains 3 buttons serving as discrete inputs to the *GarageBand* application as well, while graphically showing their functionalities in familiar forms: rewind, pause/play, and fast-forward [93].

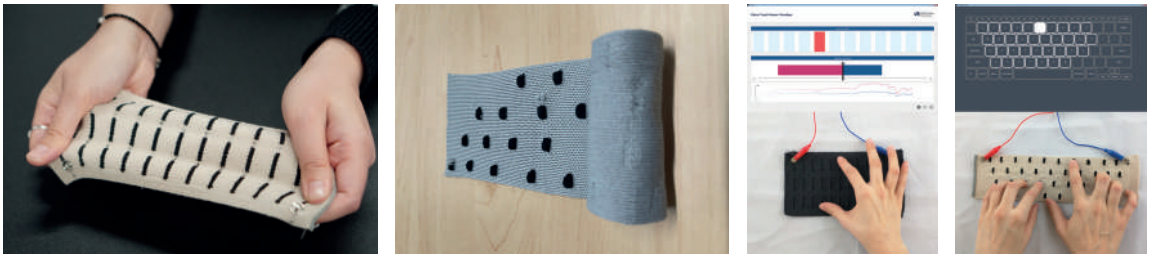
Phase 1 – Design ideas and perceptions of knitted 'Touchpad'

During this phase, the design ideas and perceptions related to the most generic sensor forms, the '*Touchpad*', were explored. Participants were shown three rectangular designs, with a set of evenly spaced touch-sensitive areas, or buttons: the *Small White Touchpad*, the *Small Black Touchpad*, and the *Large Touchpad*, whose size and specific characteristics are shown in table 6.1, and fig. 6.2(a) and fig. 6.2(b). This enabled participants to focus on understanding the potential interaction in a general sense, without a bias toward particular use cases, such as the *Piano Keyboard*, or the *Volume Controller* (table 6.1), each of which have application-specific forms. The three knitted touchpads differed only in size, or color of the non-conductive yarn, highlighting the flexibility and scalability of the production process. Each of the introduced samples was connected through two electrical wires to a visualization application (fig. 6.3(c)), as described in [132]. After the users interacted with each of the forms, I asked them to share their thoughts on the technology, and potential uses of similar sensors relying on those same principles.

Phase 2 – Usability of specialized application prototypes

During this portion of the study, participants interacted with several specialized application prototypes, illustrated in fig. 6.1 and described in table 6.1: the *Keyboard* (fig. 6.3(b)) together with an application prototype fig. 6.3(d); the *Multi-Button Controller* (fig. 6.2(c)), with its associated *Whack-a-Mole*-style game application prototype, illustrated in fig. 6.1(a); the *Slider Controller* and two game applications that can be controlled with it: *Brickbreaker* (fig. 6.1(b)) and *Flappy Bird*; the knitted MIDI controller system, demonstrated in fig. 6.1(c), and composed of the *Piano Keyboard*, the *Volume Controller*, the *Media Control Buttons* (table 6.1). These applications were selected as examples of potential use cases of capacitive knitted sensors, having real-time response, even though not fine touch location identification, developed in chapter 4 and chapter 5 of this work [132, 93]. They relied on technology introduced in [133] and their purpose was to illustrate the potential of this technology to enable interactive applications. Participants were asked to assume, while answering, that for the applications based on these sensors, touch recognition were working with high accuracy. As part of the inquiry of this phase of the study, participants were asked to think about the

advantages and disadvantages of using existing technologies versus products based on the knitted prototypes. Moreover, they were asked to express whether a product based on these prototypes would be useful and usable to them and, if so, in what context.



(a) (b) (c) (d)

Figure 6.3: Properties and applications of touch-sensitive knitted fabrics [3]: *(a)*: Stretch-ability. *(b)*: Being able to roll the fabric. *(c)*: Visualization application of the approximate location of touch on a 36-button knitted fabric. *(d)*: An application using a knitted alphanumeric 36-key keyboard as an input.

Phase 3 – Reflections on technology and suggestions

At this stage, the purpose was to understand the general impression of participants after having been exposed to different uses of this technology. Another goal was to realize if these interactions had sparked any new application ideas for touch-sensitive knitted fabrics, and which use cases had mattered more to participants. Additionally, it was important to understand: whether users would have any hesitations in interacting with these type of sensors; the aspects they would like to see improved; any potential changes to the sensors, if they were to design any of them from scratch.

6.2.3 Data Analysis

The recorded focus group audio files were transcribed using Otter.ai [5], a transcription software that converts speech to text. Following the automated transcription, the files were manually edited in two rounds for error corrections. Subsequently, the data was coded using Atlas.ti [2], a tool which helps researchers conduct qualitative analysis of textual, graphical, audio, and video data. Because analysis is based on these transcripts, in a group context, with identifiable information removed, quotations were not attributed to a particular study participant below.

Thematic Analysis (TA) [19] was selected to analyze the interview data. This approach provides

the tools for a rich characterization of qualitative data, while not placing any requirements for theory development. Its scope was considered appropriate for the purposes of understanding user response to this technology. Another researcher and I independently coded the data. Through later discussion, a consolidated and consistent list of codes and code groups was created that reduced redundancy and standardized the names. We each conducted a final round of analysis using those codes. A third researcher also reviewed code and data analysis for consistency.

I followed a pre-defined outward thematic structure of reporting findings, corresponding to the study objectives, with these three categories: the general perceptions, the potential for applications, and concerns and suggestions. In addition, there were two other major categories that emerged during the study: design aspects, and system and integration. The grouping of ideas, which was somewhat related to the three-phase design of the study, was performed this way for clarity of presentation and to respond to the study objectives more straightforwardly. However, this structure did not necessarily correspond to the way these insights emerged—for example, ideas mentioned in phase one, might be described in the section about concerns, if they are a better fit thematically.

While having previously defined major themes to explore might give the impression of a deductive analysis strategy, I argue that this approach is primarily an *inductive* one. First, the major categories were defined by the research objectives and for clarity of presentation, not by existing literature. The researchers did not have a codebook prior to analysing the data. More importantly, the information was analyzed using a data-driven approach, assigning themes based on grouping similar concepts together. The process of determining themes was iterative and flexible. The purpose was creating a thematic map which accurately reflected the dataset, and themes with internal homogeneity and external heterogeneity [11, 19, 105]. Not all themes had a similar number of data instances referring to them—the significance of themes was not necessarily dependent on a quantifiable prevalence measure, rather on whether each captured aspects of interest related to the research questions and objectives.

This analysis aims to provide *a rich description of the whole dataset*, rather than an in-depth account of one particular aspect, especially given the fact that this area has not been extensively

explored before, and participant views are less clear [19]. However, ideas of integration or concerns expressed were followed with questions about details, to better understand them, and to ensure that participants had a clear mental picture of their suggestions. A *realist theoretical approach* was employed, which assumes that language maps to experience and opinion, rather than a constructionist one, in which context and society typically play a considerable role. *Semantic themes* were used, with meaning assigned based on what the participants say, rather than looking beyond that aspect, and making inferences regarding any underlying ideas or conceptualizations. These choices were made since the purpose of this study was not to deeply explore the psychology or social context behind the participants' response. Instead, the types of technological solutions that participants would be interested in using were investigated. This way, further research can be more purposely directed towards areas of interest to users.

6.2.4 Results

This section details the study findings, starting with general perceptions of the technology, mainly based on data from the first phase of the study, but aiming to provide a holistic view of the users' sentiments. Subsequently, the desired fields of applications are described, which was the focus of the second phase of the study. Further, aspects of the knitted sensors' design as they relate to functionality and style are reported, continuing with system design and integration, also considering aspects such as device size, and interaction modalities. Finally, areas of user concern and improvements for future iterations of this technology are outlined.

General Perceptions:

- **Fabric Texture:** Participants were divided regarding their experience of the fabric texture. Many were pleasantly surprised by the softness of the fabric. One participant said: *'The fabric was a bit softer than I was expecting, because I knit a lot. And yarn that I use to knit is more coarse than that. So I was kind of expecting that type of texture. But this was softer than what I use. It was much finer than I'd been expecting when you said 'knitted''.* Similarly, another mentioned: *'You did mention, right, the threading is made out of carbon fiber, see,*

and I expected it to be a little more sharp, a little more, [...] you know [...] the opposite of fabric. [...] It's actually pleasant to see that it's not too different from the conventional fabric. So that's a nice thing'. Other participants however, considered it rougher, with one saying: *"This seems less suitable for clothing to be honest. Because it's soft, but it's not as soft as like something you would wear I think".* This sentiment was more prominent when participants were interacting with applications that required them to continuously run their fingers across it, such as using the *Slider Controller* with the 'BrickBreaker' game [93]. Another participant, while identifying the interaction surface of that same controller as *'disconcerting'*, saw that as a positive aspect, potentially useful for her autistic son to get exposed to different textures while playing a game such as 'BrickBreaker', which could shift his focus from the uncomfortable feeling.

- **Comfort and Approachability:** Overall, the fabric softness, especially in comparison with typical electronic devices, inspired users to think of applications and environments offering a relaxed and comforting atmosphere, such as hammocks at resorts, or falling asleep with your fabric electronics on your lap without risking damage to them. Additionally, it was mentioned that electronics may sometimes seem intimidating to interact with and understand, and the fabric form of the sensor might help break down the mental barrier. One participant commented: *'For me at least, fabric is just comforting in general, so especially in a hospital setting, where it's already an uncomfortable situation, maybe just introducing something to the patient that's more comfortable for them to access, would make them more prone to like alert a nurse if something's going wrong'.* Children were also mentioned in this context, for example, potentially finding it easier to communicate their feelings by interacting with their toys rather than articulating them.

- **Robustness:** Many participants mentioned robustness as a quality of the touch-sensitive fabrics introduced. *'I don't think this is going anywhere. I think this will outlive me,'* said one participant. The sensors' robustness was also considered an advantage over existing hard electronics, with one participant saying: *'But one thing definitely which beats the current*

keyboards is they are very delicate, but they will break [...] like manually. But this won't break'. This quality was also particularly mentioned for hard electronic applications related to children, who are more likely to damage them, or hurt themselves while interacting with electronics. Additionally, it inspired several participants to think about outdoor applications. One participant shared: *Yeah, so you know, put it down on the beach, and the kids can play on it, [...] can set up something that actually becomes a screen for.*

- **Portability:** The portability potential of this technology was also a characteristic mentioned in the study. The fabric can be easily folded and brought to an outdoor gathering, in the form of a game board, or a knitted musical instrument. Its light weight and flexibility allow it to be rolled and put into a bag, for example, as a keyboard for one's phone or tablet. One participant mentioned that even though a typical iPad keyboard is very lightweight, it is fragile and uses more space. Another participant said: *'If you had a business that was in gaming, and you were doing different shows, [...] this would be quick: roll it up, throw it in the back of the car, roll it back out. Then you can set up your station, and people can interact'*. In outdoor and traveling contexts, many participants would prefer applications based on these fabrics instead of existing hard electronic alternatives. For indoor environments, the answers varied by participant and situation. For example, as can be expected, instrument players would not choose fabric instruments over existing ones, even though many would use them as travelling alternatives. One participant said: *'Also, one thing to consider is that when you have a piano, it's super heavy. So it's nice to have like something that's really light, portable. [...] you could use a keyboard, but definitely keyboard and piano are not the same, right?'*. Regarding gaming, while some participants would not replace their existing controllers for indoor use with knitted alternatives, others were more interested in that idea.

- **Affordability:** Questions about cost of production were also common. Upon learning that the touch-sensitive knitted fabrics used in this study cost only a few dollars to manufacture at scale, many participants suggested several fabric-based applications as substitutes for traditional electronic ones, in cases where cost was a factor. This became especially relevant for education

purposes, or for exploring different hobbies. Fabric electronics can be used for training and learning how to perform different interactive tasks, such as playing the piano. One participant said: *'It's available to everyone. A lot of people can't afford to buy such a big keyboard and keep it. So, if you're gonna make something like this, you know, [...] if people have a hobby, they can learn it. And since this is fabric, it's going to cost way less than that.'* For a fully functioning system, the cost of the electronic components needs to be considered as well, which still is substantially lower than most conventional instruments.

Applications

- **Wearables:** Another major area that was discussed was clothing—in normal circumstances and specific occupational ones. Interactivity in controlling general functionalities such as changing music, making a phone call, or tracking physical activity were commonly suggested uses. A specific scenario that was mentioned in more than one instance was using the sensor in clothes as an inconspicuous call for help in dangerous circumstances, as expressed by one subject: *'...as a way to quick dial 911. Like you press it a certain number of times or in a certain pattern, because then if you're in an unsafe situation, you don't want to be obvious about something like that. You can do that. Even just [contact] your emergency contacts.'* Other uses included producing special suits for first-responders, astronauts, and athletes, with functionality added to make their jobs more efficient and comfortable.

- **Smart environments:** Incorporation of this sensing fabric technology into homes and vehicles for controlling temperature, lights, and music, was also an important theme. Such sensors would be able to be integrated into furniture, pillows, blankets, and car seats. One group of participants, however, expressed preference for voice over fabric for such functionality. Other similar uses were mentioned for artistic settings, including controlling effects in a theatre set. Another application area that was frequently mentioned in this context was monitoring with the purpose of activity recognition. Having a carpet composed of such fabric to detect human activity was a popular concept, especially related to security. One participant said: *'Yeah, for security, maybe if there are places that we do not want people to enter or you know, access. So*

we can just place this fabric over the entire place, so that [...] if there is any sort of reading, maybe we know that someone who's not supposed to be there, is over there.' Another mentioned a similar idea, while adding that such a carpet could replace cameras due to the lower cost. Another participant saw this as an opportunity to study areas of interest in a store based on customers' movements: *'Another application could be [...] a carpet in stores where you want to find out the layout, [...] what movement do people move? Like from what aisle to which aisle'.* Another subject suggested passenger position detection in a car: *'So even [...] if it is in the car, the car seat might have this part [...] in case of any emergency like a hit or crashing into something, the airbags come out based on the sensors, [...] where the passengers were instead of coming out from everywhere'.* Environments that rely on user identification were also considered, for purposes of personalization or security.

- **Gaming and Entertainment:** This area was of interest as well, with participants suggesting many uses. Indoor games were mentioned with interactivity integrated into larger areas like carpets, to enable dance and board games for example. One subject said: *'I was just thinking about fun - it could be set up almost like dance steps. Well, Dance Revolution or [...] dance schools. Brides sometimes will go and learn how to do a waltz or whatever, [...] do it at home'.* Outdoor activities were also mentioned as benefiting from these sensors' portability, with fabric versions of board games being a popular choice. Regarding gaming, some participants saw controllers based on these sensors as more desirable than existing ones, while others preferred the familiarity and functionality of current controllers, and would use these only for simple games. One participant observed the fact that upon touch, the pressure was detected as well, noting that it could be used in game controllers to give more nuance to each key-press. In addition, VR environments were mentioned, with one participant saying: *'I'm thinking, [...] can we do something with VR? So like [...], something [that's] not controllers. Well with fabric, it's [...] easier to put different shapes rather than monitors. And so you can have more control over environment. And it feels more real.'*

- **Education:** Several participants mentioned potential benefits to education due to several

qualities of the touch-sensitive fabric. One reason was the low cost of production of these sensors, compared to some electronic alternatives. They could be used to produce different electronic tools, such as keyboards, musical instruments, educational games, to be accessible to children in less affluent areas, or to anyone that wants to learn something new. One user said: *'You can have like a very simple piano. So just a quick question on thread itself, I would imagine it's not that expensive. So for instance, this thing is definitely a lot less cheaper than an actual keyboard. So, I would imagine it won't be that much of a leap to take the signal from this and turn into an actual interface that detects where your hand is for particular keys, and then having an application that can actually simulate a piano playing. Yeah. And so this can then be, you know, for [...] school districts that don't have a lot of money, but you still want music lessons'*. Moreover, a few participants in one of the groups also considered these sensors to be more user-friendly than regular electronics, potentially making for more interesting lessons, and even drawing more people to engage with science and technology. One user said: *'Yeah. All in all, it definitely, you know, it's much more user friendly, and makes the learning or whatever you going to do more approachable compared to, [...] more technical devices, [which] are a little bit scary.'*. Another agreed with: *'I think one of the challenges of science is that everyone just thinks about it, like, oh, how cool or how like, larger life it is. And I'm like, No, we should find ways to get more approachable, you know, encourage more people to get into STEM.'*

- **Healthcare and Assistive Technologies:** Many groups offered ideas related to this and related areas. Assistive technologies were mentioned, such as developing a lightweight braille system. In physical therapy, suggested applications included helping people with disabilities walk, by using a sensitive insole within shoes or socks to identify gait abnormalities, or similarly, learning how to properly grip objects. Applications related to posture correction were also explored, implementable by having sensors on one's chair which could detect body placement, and suggests adjustments to the users. Suggestions regarding medical use were also common, with some participants mentioning monitoring patients' movements on their hospital bed,

while taking advantage of the pressure-sensing potential of this technology. One participant mentioned: *'But again, this goes back to [...] a patient wearing something like this. So let's just say you want to [...] get information on [...] maybe the joints, [...] because maybe the patient [is] suffering from [...] arthritis [...] or joint issue, right?'* In addition, usability scenarios were very focused on getting information from patients' vitals—biorhythm, temperature, galvanic skin response, blood sugar level.

Design Aspects:

- **Unintrusive Design:** Participants described the sensor design as *'simple, predictable [...] in a good way,'* and *'blending in.'*. They appreciated the subtlety of design, which would allow the sensor to be suitable for different application, including clothing. One participant said: *'It [...] could detect touching a lot smaller, right, because I'm thinking of using this thread [...] inconspicuously on multiple surfaces, and then gather a lot of data there.'* Another area mentioned as benefiting from this quality is medicine, as expressed by one participant: *'I think, these days, when it comes to medicine, a lot of people want to focus more on non-invasive, because it's less work and also less risk to the patient and the medical practitioners'.*

- **Design Flexibility:** The fact that the yarn can be structured as required by the application was a point of interest and discussion, which enabled participants to think of different configurations for it, being aware of the flexibility of the design process. One participant mentioned: *'Currently it is [...] said [...] how it is supposed to move, but based on [...] which application we are using this for, we can change the way it has been designed so that it makes more sense'.* The scalability of the design came across from the ease with which participants would think of applications that could use a similar design, but have different sizes. Participants inquired about the distance between the sensing components of the structure. One participant mentioned that in confined spaces, such as clothing, he would prefer them to be closer together, while in larger surfaces, such as furniture or carpets, he would find larger areas of touch more useful. Participants also inquired about the type and thickness of fabric, differentiating between use cases. One participant said regarding the *'Large Touchpad'*: *'I feel like this is more*

like something I would sit on rather than something I would wear.' Personal style was also mentioned as a factor, with one participant saying: *'In the black one, I noticed that it's hard to see where you're supposed to be touching versus where the regular fabric is. [...] Yeah, I'd rather a design where you can see, where it's kind of like the white and black one, plus my aesthetic is kind of white and black.'*

System and Integration:

- **System Minimalism:** Overall, participants appreciated the reduced wiring and electronics associated with the system. One participant saw this system as an alternative to some electronics for soft robotics due to having fewer wires: *'I was thinking in the lines of soft robotics. [...] There we need to [...] conduct the electrical signals, right. And the type of things which we're working on, are very soft, like flexible. So, [...] instead of using all these wires [...] we can use this flexible fiber fabric'.* The matching small hardware is also important for portability, in addition to the fabric component, as mentioned by participants as well. Another participant suggested adding a few more connection points to increase the touch detection accuracy: *'So, potentially having more than just two contacts would be able to improve that resolution, right and more accurately tell where it is, like, get two more contacts on this side'.*

- **Integration:** An important point that was discussed during the study was how such sensors could be integrated with existing electronics, such as phones, laptops, Xbox, and PlayStation devices. One participant commented: *'You could probably hook that up to a phone easier than like a hard keyboard. So, if you're trying to [...] work from your phone, you can have this tiny fabric keyboard that can roll up easily and [...] write emails, because I know I prefer writing emails on a keyboard, not my phone's keyboard'.* Another described a use case with phone apps, which could open up many possibilities: *'An integration that could be possible with phones like [...] PokemonGo, as an accessory piece that you can [...] press buttons on, [...] maybe a bracelet or something that you could attach to your shirt, like cuff links almost, but with this fabric, to work with apps, so that you don't have to pull out your phone to [...] navigate on an app'.* In addition, participants were interested in the integration potential of several knitted

fabrics with different touch-sensitive areas, mainly inspired by the Garage Band controller set. Smart environments were mentioned, where such sensors would need to be integrated for an immersive experience.

Concerns and Feasibility

- **Touch Detection Accuracy:** As expected, it became clear from almost every group that in order for users to adopt this technology, it had to work accurately and reliably, similarly to other objects and applications it would emulate. Questions of multi-touch detection were also raised, and suggestions were made to add more contact points, by adding another yarn. In addition, multi-touch functionality was mentioned as desired or expected.

- **Damage and Safety Concerns:** Besides accuracy, safety was a concern in many cases, including the sensors' flammability, their electromagnetic field, and generally the participants' perception of having electronics so closely integrated into clothes. One participant asked: *'Is there like a chance, like any remote possibility of an electric shock?'*, and similar questions were not uncommon. Another participant was concerned about such issues caused due to interaction with water: *'Well, I worry about the wet. I throw it on the grass and my grass has dew on it, [...] and somebody's feet zapped, or their hands zapped. Or is it going to short out the electronics that it's connected?'* Others worried about maintaining the integrity of the electronics and any potential fires, with one participant inquiring about *'melting because the heat on a polyester material'*. Another reason for hesitation was also the potential for the sensor to easily malfunction, while offering no straightforward ways to troubleshoot, with one participant commenting: *'My question would be troubleshooting. [...] So usually [if] my electronics stop working, I can troubleshoot and kind of try and come down to the solution on my own [...] I'm not even sure how you would begin troubleshooting. [...] So kind of just like that idea of if something goes wrong, I don't know how to fix it'*. A few participants were also concerned about any potential waves emitted from the electronics, and their effects on their health, with one participant asking: *'If you are going to use it for clothing, [...] is there any kind of emissions [...] from the wires inside?'*

- **Everyday Use Reliability:** *'What if it gets wet?'* was one of the most frequent questions, and related inquiries included *'spillages like coffee or hot water'*, and exposure to saltwater, sweat, and rain. One participant suggested coating the yarn with water-resistant material. A related concern is that of false positives, with a similar effect to touch, possibly caused by any conductor in contact with the interactive areas of the sensors. Participants also asked about the sensor's capacity to only respond to intentional human touch. They were also interested to know whether these sensors were robust enough to withstand differing conditions such as stretching, and laundering, while retaining their properties: *'Can I throw it in the washing machine and dryer, if it's a blanket? Or can I use industrial cleaner on it? If it's in my car?'*, inquired one participant. Another asked about the limits of their working and their breaking point: *'How about long term reliability? I don't really know that much about like, carbon fiber and its properties, but for instance, if you were to like fatigue test this and fold it many, many times, [...] eventually cause problems'.*

6.2.5 Summary and Discussion

The results of this formative study indicate considerable interest in furthering this technology, with participants proposing and being interested in a variety of usability scenarios. Participants' general perceptions of touch-sensitive knitted fabric were that this technology is comforting, portable, robust, and affordable. There were discussions regarding its texture, which was also a point of interest, and should be considered during application design.

Many applications were mentioned during the discussions, with the most major areas being: clothing, smart environments, gaming and entertainment, education, and healthcare and assistive technologies. Prior work has investigated applications of smart textiles in some of these areas, such as smart homes [152], healthcare [12, 13, 75], wearable devices [144, 153], and accessibility [137, 21]. While users in this study were presented with the basic working principles of this technology, they were not experts in the field, which led to them occasionally proposing applications that are not directly possible using this technology in its current state. Examples include medical applications of gathering information about patients' vitals, which was an idea suggested by several users. The

functionality of these sensors is touch detection, and that is dependent on the type of yarn and circuit design. However, this tendency of the participants is worth noting for future developments, and for fabric-based sensing that might rely on other technology. Other work in fabric sensors has investigated such functionalities, such as heart monitoring [12], and joint moments [13] among others.

Similarly, many applications about monitoring were mentioned, which could be feasible, however it should be noted that these knitted sensors respond to the proximity of a conductor, which could be human skin, conductive shoes, or gloves. Another notable aspect of discussions regarding applications was that many of them would rely on the functionalities of pressure sensitivity and gesture recognition, among others. Section 6.3 of this work starts exploring this technology's potential for gesture recognition. Pressure sensitivity should be investigated in future work, since it also holds considerable promise for enabling applications.

When asked, users noted several points of hesitation when interacting with this technology. As expected, touch detection accuracy was one of them, in addition to multi-touch detection. One user suggested adding more conductive yarn to increase the sensor's accuracy. In order to accomplish that however, circuit design compatible with digital knitting, and its current flow without shorting need to be ensured. Nevertheless, advancements in accurate touch detection and multi-touch are necessary for the proper functioning of the sensors, and work in previous chapters of this dissertation has started addressing accurate touch location identification on them. Future work should further those efforts, and should additionally, investigate multi-touch location identification.

Users also raised sensor damage and safety concerns. Presently, the touch sensing hardware operates at both a low voltage and low current and does not emit significant electromagnetic radiation. Cursory experiments involving moisture have shown a potential decrease in touch location accuracy, but no apparent harm when touched, however further experiments should rigorously investigate those aspects. Additionally, participants discussed everyday usage robustness, such as the sensors' exposure to water, rain, sweat, or their ability to resist damage from stretching, washing and drying. These are understandable concerns regarding the real-world scenarios to which such sensors

would be exposed, if they were integrated in interactive applications. For comprehensive answers to these questions, specific applications and their needs should be considered as well. The sensors are expected to behave differently due to exposure to water, especially if it contains electrolytes, such as in the case of saltwater, sweat, and rainwater. The level of moisture is also expected to impact the sensor's conductivity. Electrolyte solutions are conductive, and they can change the current flow within the circuit, therefore it is important for future work to thoroughly investigating those aspects. Section 6.4 starts addressing some of everyday use concerns, specifically related to normal laundering and stretching of the sensors.

6.2.6 Application Design Guidelines

Despite any hesitations or points of discussion, the participants' enthusiasm to interact with this technology and desire to use real-world applications based it, opens many opportunities. Based on these results, I offer some guidelines regarding building interactive applications with touch-sensitive knitted fabric.

- Participants cared about the texture of the fabric, including the carbon-coated interactive components, especially when interacting with them to control an application. Some considered it soft, others not soft enough to use, and there were also views of it not being soft, but proper for some applications because of that quality. Therefore, when deciding the patterns of the fabric and related application functionalities, application designers should account for the sensation that the continuous contact with the interactive components generates. This consideration should be in accordance with the target user group. In addition, future research should explore new, softer conductive yarn materials to offer more flexibility in application design.

- Participants perceived touch-sensitive fabrics as compatible with relaxed environments, or capable of generating feelings of approachability and comfort. This psychological aspect should be leveraged when designing applications to add an extra dimension to the user experience. Since fabric is integrated into everyday objects, such sensors could be useful for creating inviting

smart environments.

- The flexibility of the design process allows this technology to be suitable for a large variety of situations and use cases, about which users care. The importance of personal style was noted in this study, and it has already been established in literature [56, 44, 30]. Therefore, users should have a variety of choices, at least regarding colors, sizes and patterns to match their preferences. Such choices can be easily accommodated through the streamlined manufacturing process that requires little-to-no human intervention.

- Other related considerations for application designers should be the fabric' size, its thickness, and the balance between flexibility and sturdiness, as they relate to the applications for which the sensor is designed. For example, users would probably prefer to wear fabrics that are flexible and choose sturdier ones for carpets. Similar considerations have also been investigated in prior work [102, 94].

- These sensors' low profile and unintrusive design could be an important aspect in applications that would benefit from discreet interactions.

- In addition to novel applications, or applications that replace existing technology, fabric sensors can be developed to be used alongside existing technology, but in different contexts. Examples include fabric musical instruments useful for travelling or practice, and relevant due to qualities such as portability or affordability, but not replacing those instruments.

- Virtual and augmented reality (VR/AR) applications are a direction which should also be explored, since touch-sensing knitted fabric can provide more control over the environment than regular controllers. Such fabrics can be the basis of constructing entire environments, since the shapes of both objects and their interactive components can be flexible, and easily modified. Due to the fabric softness compared to hard electronics, such an implementation could be a safer alternative, since in these settings, especially in VR, users might be less aware of their real environments.

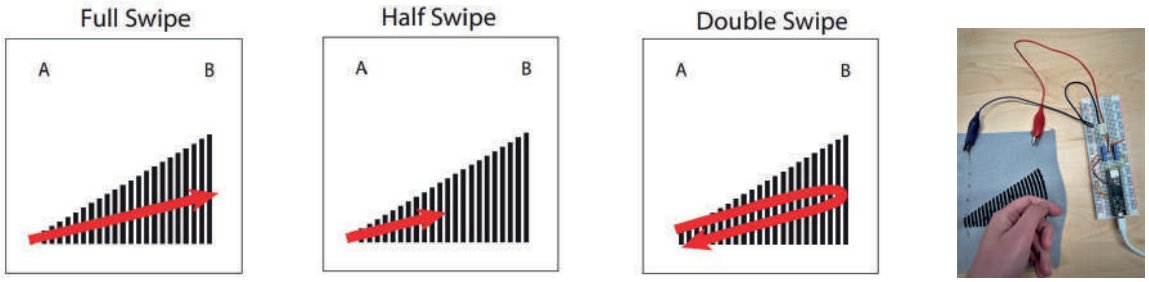
- Additionally, children and the elderly could be particularly suitable target groups for interactions relying on this technology. The sensors' softness and robustness could offer protection from accidental harm. Previous work, for example, has explored playable surfaces for autistic children [100, 137, 21].

- These sensors were specifically designed to have the smallest number of connections that could create a circuit, in order to increase their usability and robustness [133]. Further research should investigate the necessary number of connections per design pattern and application type. However, the ability of sensors produced using this technology with minimal wiring and connections, to easily connect to each-other, as well as external hardware, is an important feature which could facilitate their adoption. Future applications should keep easy integration and modularity at the forefront of their design strategies.

- Since damage and safety concerns were raised from the participants, once all have been properly investigated and addressed, they should also be clearly communicated to the end users. It is important not only for the technology itself to be safe, but also for users to perceive it as such and be able to trust it, in order for them to freely interact with it and integrate it as part of their everyday environments.

6.3 Gesture Representation

One of the main capabilities that this technology has the power to enable is gesture recognition, which was also mentioned as expected or desired by the users in the formative study above. This section explores the representation of swipes on the *Volume Controller* sensor (fig. 6.4), designed to be intuitive for such gestures. Three related swipe gestures are included, as well as a fourth class of gestures to represent accidental touch events during everyday use. In order to investigate the quality of the generated signal, I compute the similarity between different gesture representations by computing the pairwise distance between all samples, as in section 4.7 [132] above. In order for this technology to show promise and feasibility, different samples of the same gesture type should be more similar to each-other than signal samples produced by different gestures. Additionally, the



(a) (b) (c) (d)

Figure 6.4: Experimental setup including the swipe gesture illustrations which participants were shown, corresponding to the gestures categories: *(a):* full swipe gesture, *(b):* half swipe gesture, and *(d):* double swipe gesture. An example of the last category, random contact, is shown in *(d)*, together with the data collecting hardware. Gestures varied for that category, since participants were not shown a particular motion trajectory, as in *(a)*, *(b)*, and *(c)* — some examples were mentioned to them, and they were informed of the types of situations that might generate accidental touch events. These illustrations were produced at CFF [3].

signal response of accidental touch events should be different from that of intentional gestures.

6.3.1 Experimental Procedure

To investigate if basic gestures can, in future iterations, be distinguished on this sensor, data from 12 users was collected. The sensing hardware and signal processing method are the same as those described in section 5.2.1. The hardware uses an NXP Kinetis® MK66 ARM® Cortex®-M4F180 MHz microprocessor (PJRC Teensy 3.6 development board). Bode analysis [18], a system identification approach used to characterize the behavior of an unknown system, given controlled inputs, is also used for characterization of the signal, as in section 5.2.1. Two 16-bit analog-to-digital converters (ADCs) synchronized with the DAC sample the output waveform at approximately 256k samples-per-second in 1024-point increments. Fast-Fourier Transforms of the samples were processed using the ARM CMSIS DSP library complex FFT function [72]. Ninety six frequencies between $f_0 = 2,000$ Hz and $f_1 = 26,000$ Hz were input for a duration of $t_0 = 0$ seconds to $t_1 = 0.004$ seconds. To characterize the system response during data analysis, the gain values of all recorded frequencies are used, for a total of 192 values per record window, which is 250 time-steps long.

Participants were asked to perform four different types of gestures: *full swipes*, *half swipes*, *double swipes*, and *random contact*. Each participant was asked to perform 20 repetitions of each of the

mentioned gesture with a single finger, but with varying speeds (max 4 seconds) and pressure. These gestures are illustrated in fig. 6.4. For the random contact gesture class, participants were asked to imagine the sensor integrated into clothes or furniture, and perform gestures to depict accidental touch while interacting with them in daily life. In this case, multiple location and finger gestures were collected. Some examples of random contact gestures are: multiple taps, brushing across the sensor, or resting one's palm, wrist or elbow on the sensor. A total of 240 samples were collected for each of the 4 classes from all participants.

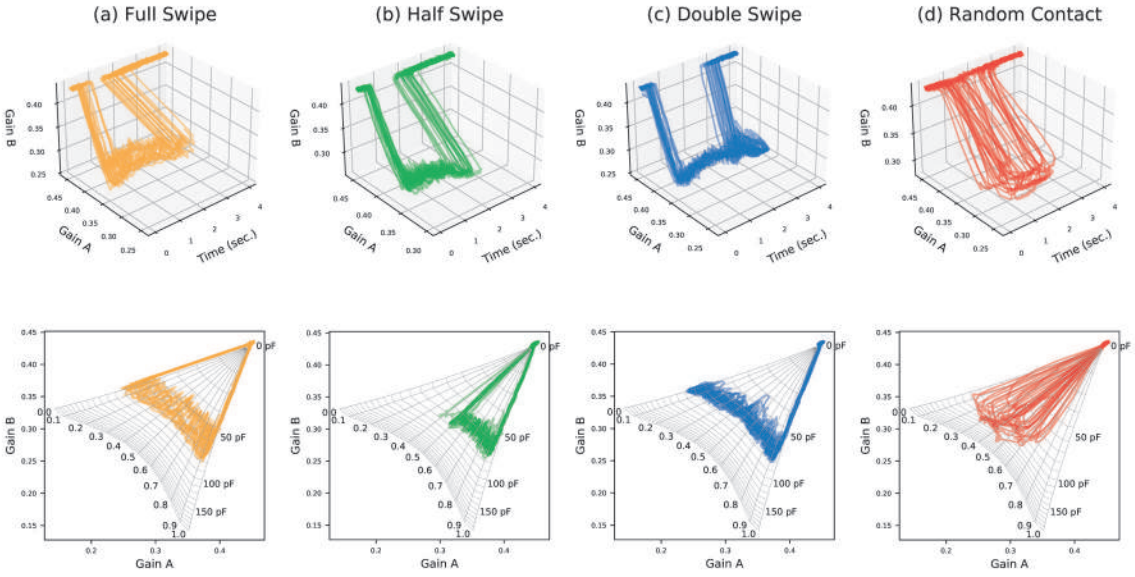


Figure 6.5: The representation of a swipe gesture using voltage gains from two signals over time. Each plot column contains 20 instances of swiping across the volume controller to perform the same gesture, obtained from one participant. The top row visualizes the A and B voltage gains over a 4 second duration. The bottom row contains a flattened view of the A and B voltage gains throughout the duration of each gesture overlaid on a grid depicting the linear location and capacitance mapping. These plots illustrate three similar but different swipe gestures and accidental touch events: *(a)* a single swipe across the volume controller, *(b)* a swipe that stops halfway across the controller, *(c)* a double swipe, starting by the swipe in (a) and then continuing in the reverse direction of it, *(d)* contact emulating accidental touch events. These plots were produced in collaboration with CFF [3].

6.3.2 Data Analysis

The plots in fig. 6.5 illustrate gesture signals captured as explained above. The data was collected from one participant performing the four different gesture types, 20 times each, and is meant to provide an intuitive understanding of the trajectory of the signal expressed as two gain values over

time. We can see that samples of the same gesture type follow a very similar path, more clearly shown in the top plots. The bottom plots illustrate how different gestures follow distinct trajectories from each-other. Even random contact samples, emulating accidental touch events, seem to be more similar to each-other than to the swipe gestures.

Next, I perform a comprehensive analysis of all the samples in the dataset. In order to measure the similarity relationship of different gesture samples belonging to the gesture type, as compared to samples that belong to different gestures, the pairwise distance for every sample pair in the dataset is computed, similarly to section 4.7. A 4×4 matrix is generated, where both rows and columns indicate the type of gesture: *single full swipe, half swipe, double swipe, and random contact*. Each gesture sample is associated with its label, and when two samples are compared to each other, a scalar value that represents their distance is produced. That value is added to the matrix in position (m,n), where m and n denote the label of the first and second samples being compared, respectively. I use the Euclidean Levenenshtein Distance (*ELD*), introduced in section 4.5, as a distance metric, in order to calculate the similarity between two gesture samples.

ELD is a distance metric which is a modified version of Levenshtein Distance [82] and uses a similar concept to it in that it performs a pairwise comparison of two sequences, in this case consisting of gain values in a gesture sample. Within a gesture sample, the gain values are temporally related and the voltage discharge during touch is a process with a relatively pre-defined trajectory, therefore each gesture sample can be considered a sequence. If two gesture samples are compared sample-by-sample, the resulting *ELD* is a measure of similarity between them. A smaller distance indicates higher levels of similarity, since converting one gesture sample into the other would cost less.

6.3.3 Sample Similarity Results

The generated heatmap, illustrated in fig. 6.6, has a size of 4×4, where each cell in either direction corresponds to a gesture type. Each element of the heatmaps' diagonals is composed of the sum of such distances of different samples of gestures of the same class, while the rest of the elements result from sums of distances of gestures of different classes. It can be noted that the diagonal, which reflects distances of gesture samples of the same type, is composed of lower values compared

to distances of samples of different swipe gestures. This means that there is less similarity among different-type gesture samples and more similarity among samples belonging to the same gesture. Additionally, I compute some statistical measures on the resulting distance matrix to gain more insight into what the data means. Two groups are investigated:

1. The distances between same-type samples, encoded in the heatmap diagonal.

2. The distances between different-type samples, encoded in the rest of the heatmap.

The *f-statistic* and its associated *p-value*, p_f, resulting from a *one-way ANOVA* are recorded, as well as the *z-score*, and its associated *p-value*, p_z. These inferential statistical tests measure separability of groups. The lower each *p-value* is, the more significant are the group differences considered. A *p-value* ≤ 0.05 is generally accepted to mean that the difference between the groups in the data is statistically significant. Results in table 6.2 show that both p_f and p_z are equal to 0.00, which means that there is an estimated 0% probability of the differences between our two groups having occurred by chance. These values, together with the heatmap in fig. 6.6, suggest that the gesture representations of the same type are similar to each-other and different from samples produced by gestures of different types.

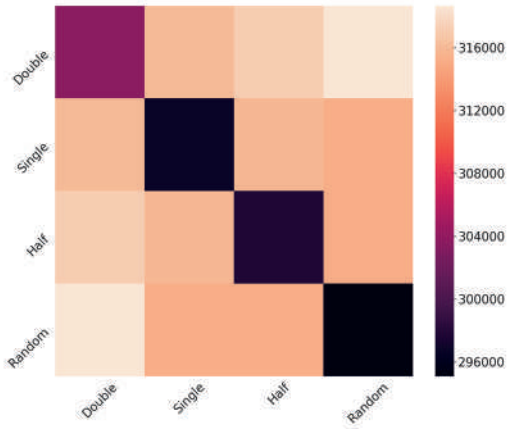


Figure 6.6: Heatmap indicating gesture distances, computed using the sum of *Euclidean Levenshtein Distance* of all gesture sample pairwise comparisons. Gesture samples of the same type are more similar to each-other than samples from different gesture types.

Table 6.2: Statistical significance of similarity matrix results. These values suggest that the difference between group *(1)* and group *(2)* comparisons is significant.

$f-stat$	p_f	$z-score$	p_z
231.81	0.00	-15.23	0.00

These results are encouraging since it seems, it is not only possible to distinguish among different types of swipes, which are gestures closely related to each-other, but also accidental touch events. From fig. 6.6, it can be seen that even gestures in the *random touch* class have the potential to be categorized together in future applications. For those gestures, as mentioned, participants were not specifically directed to trace a particular trajectory, but to emulate accidental touch events on the sensor. These results hold promise for creating gesture recognizing applications built on the basis of this technology. However, since these sensors would need to operate in the real world, they should be able to resist different types of potential distortion. The section below starts to explore this aspect.

6.4 Robustness to Distortions

In addition to creating models for high-accuracy touch location identification, exploring gesture recognition, and users' views regarding these sensors, it is also important to understand how robust they are when exposed to possibly disruptive everyday conditions. Sources of distortion in real-world environments can be many, however, in this section two are explored: sensor laundering and stretching. They were selected as intrinsically tied to the nature of fabric and its functionality.

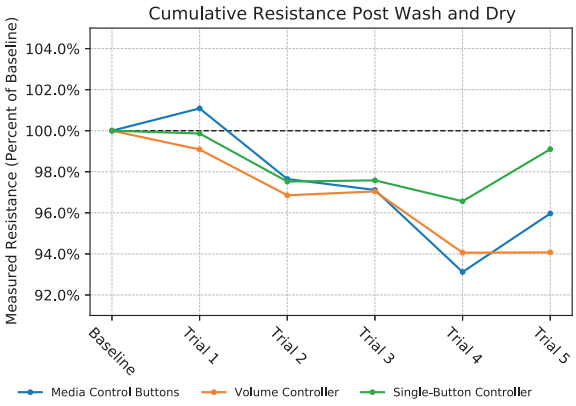
6.4.1 Washing and Drying

Many applications for which these sensors were designed, and many of the ones discussed in the focus group above are wearable, which means that the fabric component of each sensor would need to be washed, for appropriate use in real-world settings. The integrity of the sensors after undergoing normal washing and drying was a concern that participants also mentioned. Several washing and drying tests were conducted to study any potential changes in resistance of the knitted sensor, which would affect their properties.

Methods

Three sensors were tested, illustrated in fig. 6.7(a): the *Volume Slider*, the *Single Button Controller*, and the *Media Control Buttons*. These sensors were selected to provide variety regarding their size and the structure of their interactive components. They were washed according to the American Association of Textile Chemists and Colorists (AATCC) Laboratory Procedure 1-2018 Home Laun-

(a)



(b)

Figure 6.7: In *(a)*, we can see the samples that underwent 5 cycles of washing and drying. In *(b)*, the change in resistance values of those sensors across all cycles is shown. The y-axis values are calculated as the rate between the trial and baseline resistance values. These results demonstrate that there is only a slight decrease in the resistance values after undergoing washing and drying [3].

dering: Machine Washing protocol [1]. It specifies a 35–minute wash duration with 1.8 kg of laundry and 66 ± 1 g of detergent, and a standard tumble drying protocol, with a $68 \pm 6°C$ temperature. This protocol was considered appropriate for everyday laundering of clothing.

Five cycles of washing and drying were completed using an industrial On-Premise Laundry (OPL) Wascomat WUD718cc washer, a Purex liquid detergent, and a Wascomat D735 dryer. It is worth noting that these sensors had been washed and dried before these experiments, in addition to being steamed, as part of their manufacturing process. Baseline resistance measurements (b) were recorded before the testing began. Following this, measurements were taken after each washing and drying cycle d_n (where $n = 1$ to 5 cycle number), resulting in 6 measurements in total per sensor. For the *Media Control Buttons* sensor, measurements were also taken between components of the sensor, in order to obtain a better understanding of the internal sensor structure integrity. For each test, the percent change in resistance between the baseline measurements and measurements after each washing and drying cycle was calculated as $\%\Delta R_{(b,d_n)} = (R_{d_n} - R_b)/R_b$. In this case, R_{d_n} stands for the resistance value between the endpoints of the sensor after the n^{-th} washing and drying cycle, and R_b for the baseline resistance value between those same endpoints, before any of the washing and drying cycles recorded. For each measurement, 100 samples were collected using a Keysight

34465A digital multi-meter, and their mean is reported as the measurement value.

Results

Table 6.3: The percent change in resistance $\%\Delta R$ between the baseline b and each test d_n is reported. The resistance is measured between the two yarn endpoints of each sensor. These results show that the resistance values after each washing and drying cycle are within 7% of the baseline.

	$\%\Delta R_{(b,d_1)}$	$\%\Delta R_{(b,d_2)}$	$\%\Delta R_{(b,d_3)}$	$\%\Delta R_{(b,d_4)}$	$\%\Delta R_{(b,d_5)}$
Single-Button Controller	−0.14%	−2.48%	−2.42%	−3.44%	−0.90%
Volume Controller	−0.92%	−3.15%	−2.96%	−5.94%	−5.93%
Media Control Buttons	1.09%	−2.35%	−2.88%	−6.88%	−4.03%

Results from the washing and drying trials of all three sensors are included in fig. 6.7(b) and table 6.3. Figure 6.7(b) shows how the resistance changes from the baseline after each washing and drying cycle for the three sensors in this test. There is a slight downward trend of the resistance between the first and last trials. Table 6.3 calculates the percent change in resistance between each trial and the baseline, with changes ranging from approximately 0% to 7%. More extensive tests should be run with such sensors to quantify the amount of change in resistance due to laundering, and subsequently those changes need to be incorporated into the models which enable their fundamental working for later use in interactive systems.

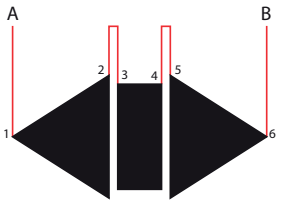


Figure 6.8: Sketch of the *Media Control Buttons* sensor, annotated with points along which the resistance was measured. The red lines show the unexposed carbon-coated yarn in the textile.

In addition, fig. 6.8 and table 6.4 show the change in resistance between different points in the sensor, calculated between the *baseline, b* test and the last *washing and drying test, d*5. The visible interactive components in the *Media Control Buttons* are designed with the conductive yarn knitted closely together, and between the components, there are connections running within the fabric substrate, which are also part of the circuit. Measurements were taken from point A, an endpoint of the carbon-coated yarn, to every point where the conductive yarn enters $(p1, p3, p5)$ or exits $(p2, p4, p6)$ an interactive component, until reaching the other yarn endpoint B. The reported results show a cumulative effect of the resistance of the hidden and visible

conductive components from A to B. The resistance measures indicate more change in the unexposed conductive yarn connections compared to the rest of the components, possibly due to them being more resistive than the rest of the components, since the current path is not as widely spread as the bigger interactive components. Knitting a bigger area with conductive material decreases its resistance, allowing more current to flow through it. Further studies need to investigate these relationships to predict the sensor's and its components' behaviour after several cycles of washing and drying. Nevertheless, the most prominent changes reported here are still within 6% compared to the baseline, which is in line with the results from table 6.3.

Table 6.4: The percent change in resistance ($\%\Delta R$) values between the *baseline (b)* and the *last washing and drying (d5)* test. These results are reported for measurements across and between visible sensor components.

	$[A, p1]$	$[A, p2]$	$[A, p3]$	$[A, p4]$	$[A, p5]$	$[A, p6]$	$[A, B]$
$\%\Delta R_{(b,d_5)}$	-5.81%	0.20%	-2.31%	-1.66%	-5.75%	-3.29%	-4.03%

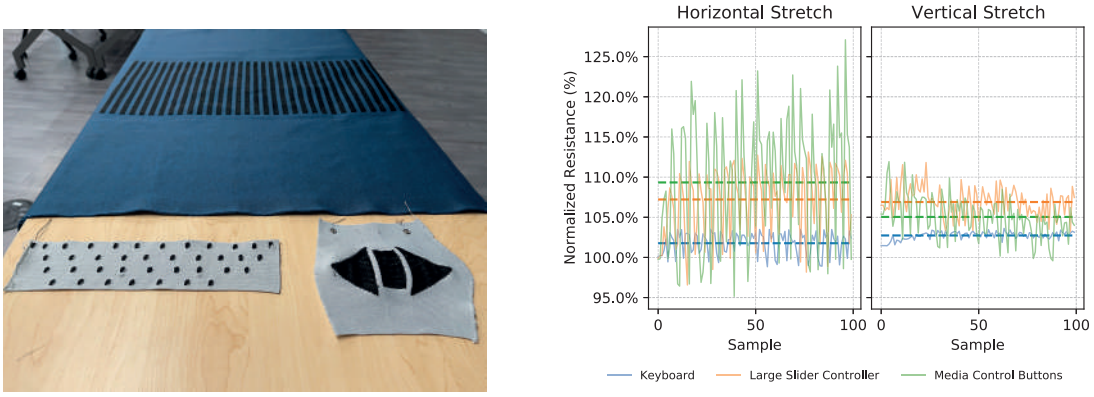
Together, these results demonstrate that there are slight changes in the resistance values of the sensors, due to washing and drying, but these changes are bounded and predictable, indicating sensor robustness. They should be taken into consideration however, when creating location identification or gesture recognition models, such that interactive applications can be reliably built upon them.

6.4.2 Sensor Stretching

Stretchability is another inherent property of knitted fabric, therefore its effect needs to be considered on the knitted sensor circuits. This experiment, similarly to the above, considers the change in resistance due to a potential distortion, which, in this case is stretching of the sensor. Two cases are investigated: vertical V, and horizontal H stretching of the sensor. Additionally, a baseline state b is measured, while the sensor is not being stretched.

Methods

Similarly to the experiment above, for each measurement, 100 samples are recorded using a Keysight 34465A digital multi-meter and their mean is reported as the measurement value. For this experiment, three sensors were used, illustrated in fig. 6.9(a): the *Keyboard*, the *Media Control Buttons*,



(a) (b)

Figure 6.9: In *(a)*, we can see the samples that were used for the horizontal and vertical stretch tests. In *(b)*, the change in resistance values of those sensors over the 100 consecutive collected samples is shown. The solid lines show the resistance values over time, while the dotted lines, the average for that sensor's samples. The y-axis values are calculated as the rate between the sample and baseline resistance values [3].

and the *Slider Controller*. For each test case, the sensor was manually stretched in one direction and allowed to revert back to its natural position. This happened continuously over the 4 seconds during which the 100 samples were being recorded. Figure 6.9(b) shows the resistance values during the horizontal and vertical stretch test for the three samples, and table 6.5 summarizes it, by reporting the maximum and average change in resistance during stretching as compared to the baseline, calculated over the 100 sample points per test.

Results

The changes observed depend on the shape of the sensors and the carbon-coated yarn pattern. From these results, we can see that the changes in the *Slider Controller* were the most consistently low, possibly due to the fabric's large size and thicker, sturdier construction. The two smaller, thinner samples showed more variability, with the maximum change in resistance being the *Keyboard* sensor stretched horizontally, at 27%. While these samples get stretched, the structure of the circuit changes as air gets in between the yarn loops, affecting current flow across the conductive areas, which could explain these results. Nevertheless, future work should explore these aspects more. Also, more rigorous stretch testing, such as incrementally increasing the amount of force applied and measuring

the change in resistance, would help clarify this phenomenon.

Table 6.5: The percent change in resistance $\%\Delta R$ between the baseline and each of the two stretch tests (*horizontal, H* and *vertical, V*) is reported. Figure 6.9(a) shows the three sensors used for this experiment. The resistance is measured between the two yarn endpoints of each sensor. The maximum and average changes in resistance are reported.

	$\%\Delta R_{max}(V)$	$\%\Delta R_{avg}(V)$	$\%\Delta R_{max}(H)$	$\%\Delta R_{avg}(H)$
Slider Controller	3.61%	2.73%	3.54%	1.77%
Media Control Buttons	11.81%	6.90%	13.12%	7.21%
Keyboard	11.92%	5.04%	27.09%	9.34%

6.5 Conclusion

The work in this chapter focused on exploring user perceptions of touch-sensitive knitted fabrics in general, and knitted capacitive sensors constructed using one conductive yarn and two external connections in particular. By conducting a formative study and performing thematic analysis on the collected data, I summarized users' sentiments, discussed potential applications and their feasibility, investigated the relationship between choices in design patterns, system construction, and expected sensor functionality, and explored user concerns and suggestions. These results indicate that there is considerable potential for this sensing technology to enable interactive applications and be seamlessly and organically integrated into users' lives. Subsequently, based on these findings, I offer several suggestions for future interaction design based on touch-sensitive knitted fabric. Furthermore, some salient user concerns related to the sensors' performance in the real world started to be addressed through another user study and two experiments.

The ability of the *Volume Controller* sensor to represent simple swipes with high fidelity was investigated. Experimental results indicate that it is possible to differentiate among different gestures, as well as accidental touch events, which are common in daily life. They demonstrate that gestures of the same type are more similar to each other than gestures of different types, indicating potential in building gesture recognition systems on these sensors. Two more experiments measuring the sensors' resistance values were conducted: a washing and drying test, and a stretching test. The sensor resistance value is related to its circuit design and affects the data that can be output from the sensor. A stable resistance value is an important factor for building a reliable sensing system.

Results from both experiments indicate robustness in the resistance measurements.

Several aspects of this technology need to be explored and developed, including accurate multi-touch and gesture recognition on any sensor design pattern, which is a considerable undertaking. More experiments need to be conducted while these sensors are being used as intended by different applications in the real world. Further qualitative user studies need to address more focused aspects of the design, integration, and their usability potential. Nevertheless, through the user studies and experiments presented here, we have moved closer to having knitted smart sensors ubiquitously integrated into everyday environments, similarly to regular fabric. The next chapter finalizes the contribution of this work by more fundamentally exploring the construction of a high-accuracy gesture recognition system, upon which many applications can be constructed.

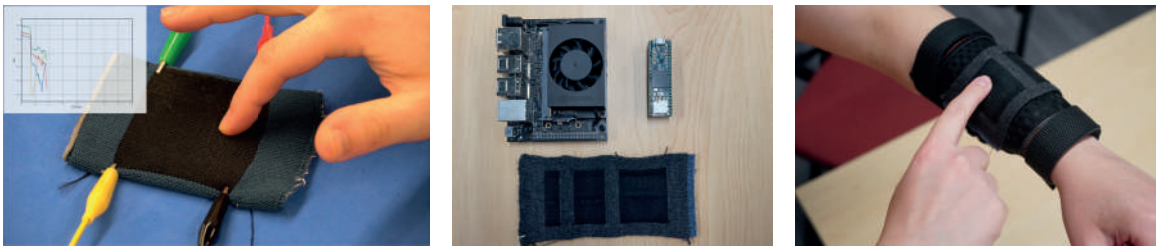
Chapter 7: Gesture Recognition

7.1 Introduction

Innovation based on fabric sensors has the potential to enable many interactive applications. Through machine learning, the work in this chapter advances gesture recognition in capacitive knitted sensors, designed to rely on one conductive yarn and few external connections. The ultimate goal in developing the underlying technology of these sensors is allowing them to behave like fabric in the real world [40], while producing accurate real-time outputs that model complex behavior, upon which it is possible to develop interactive applications.

The recognition model relies on a combination of Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) neural network architectures to capture the important aspects of the signal generated from different gestures. CNNs learn local relationships, while LSTMs focus on the sequential aspect of the time series signal data. This work, in chapter 4 and chapter 5, has explored accurate touch location identification on a 36-button knitted touchpad, which uses the same design principles and construction process. However, those systems still do not offer gesture recognition capabilities. In addition, chapter 6, using the *ELD* metric, started investigating the capability of this technology for gesture recognition, and showed that it is possible to differentiate among gestures of different types on a knitted sensor relying on one conductive yarn. However, first, the gestures explored were only three different swipe types, in addition to gestures emulating accidental touch events, and second, they were relatively simple. Moreover, no gesture recognition was performed in the chapter 6 exploration.

This chapter builds the foundations for creating an *interactive gesture recognition system*, with the potential to enable many applications. A *sensor pattern* is introduced, which is designed through programmatically routing the carbon-fiber yarn for gesture input, containing one solid sensing area and 4 electrodes to connect to external hardware. I present a *supervised CNN-LSTM neural network model* to classify 12 relatively complex gestures performed on a knitted sensor, which are a subset



(a) (b) (c)

Figure 7.1: Knitted sensors for gesture recognition, constructed with carbon-coated nylon yarn (darker rectangular regions) and polyester yarn [3]. *(a)*: A knitted sensor with four electrodes attached capturing gesture information. *(b)*: Components to construct future real-time gesture recognition systems: an NVIDIA® Jetson Xavier™ NX Developer Kit hosting a trained model, a micro-controller for signal generation and processing, and the knitted sensor for gesture input. The knitted component is a similarly designed and constructed sensor as in *(a)*, but with three smaller conductive areas. *(c)*: The same knitted sensor shown in *(b)* being worn.

of the characters of English language. In addition, this chapter introduces and explains *results from three user studies* for training, evaluation of the model under normal lab conditions, and evaluation of the model while the sensor is being worn. Finally, to get closer to real-world use, *results from an experiment* investigating the effect of washing and drying on the sensor's resistance are presented. The resistance of a sensor is an important property related to its circuit design and ultimately trained machine learning model. These advancements demonstrate the viability of low-profile, flexible knitted sensors for real-world use, through a gesture recognition model offering 90% subject-independent accuracy, while chance is 8.3%, and lightweight, portable hardware for deployment. Through empirical evaluation, this system shows robustness under some potential sources of distortion.

In the section below, the gesture recognition system is introduced, composed of the knitted component, signal measurement circuit, signal processing pipeline, the gesture recognition model, and the deployment hardware. The user studies, in the successive section, serve to collect data for training the model, and to evaluate its stability with new data. Next, the applicability of this work in the real world is explored, by furthering its technical evaluations and discussing the necessary processes and components to build applications relying on the gesture recognition capabilities of such a system. Following these explorations, the remaining sections provide more context, limitations,

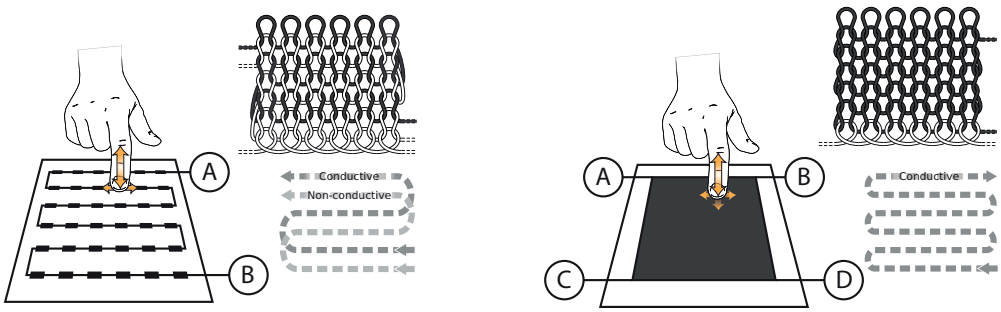
and future directions.

7.2 System Design

This section describes the gesture recognition system, composed of a few main components: the knitted fabric containing conductive and non-conductive yarn, the measurement hardware and circuit, the algorithmic components which produce a trained machine learning model, and ultimately the NVIDIA® Jetson Xavier™ NX, a powerful embedded system-on-module (SoM) on which the model is deployed. Currently, signal generation, acquisition, and processing occur offline through the use of an arbitrary waveform generator and digital oscilloscope. After the data from multiple users is collected, I train a classification model to distinguish among 12 different gestures. Subsequently, the model is deployed on the embedded CPU and its ability to recognize a gesture event in real-time is tested. In the future, signal generation, acquisition, and pre-processing should be performed on a standalone embedded micro-processor, capable of communicating with the embedded CPU, which would allow this system to be fully portable, while providing real-time interaction.

The selected gestures are the alphanumeric characters: *'3'*, *'5'*, *'I'*, *'J'*, *'L'*, *'M'*, *'O'*, *'S'*, *'V'*, *'W'*, *'Z'*, with the addition of character *'?'*. All of these gestures can be performed in one motion and have a distinct onset and offset. Some are more basic, such as *'I'*, others more complex, such as *'?'*, and some gestures are more similar to each other, such as *'5'* and *'S'*, which could cause incorrect classification. This character subset contains the basic motions upon which other alphanumeric characters can be built. This choice of several alphanumeric characters serves to strike a balance between prototyping this system, and the complexity it has the potential to offer. Instead of testing its recognition ability with more basic gestures, such as taps and directional swipes, a more comprehensive set was selected, including gestures with curvatures, to explore the limits of its capabilities. On the other hand, including the whole alphanumeric character set of the English language would increase its complexity even more, requiring more training data. It is worth noting that the focus of this work is not the specific application, but the gesture-enabling technology illustrated through some examples. Ultimately, a recognition model capable of detecting numbers and the letters of the alphabet could be built, which could serve as a communication system.

This system is a first step in that direction, but more generally, it unlocks the potential of gesture recognition on knitted sensors with many possible applications. Below its components are described in more detail.



(a) (b)

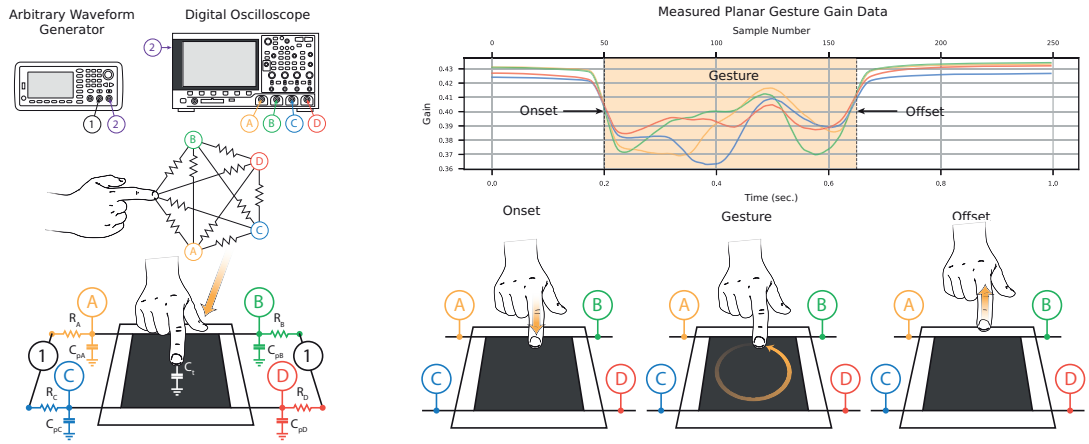
Figure 7.2: Diagrams of the knitted touchpad designs, produced at CFF [3]. *(a)*: A single-wire touchpad created using conductive and non-conductive yarns that serpentine across the textile surface. The points *A* and *B* connect to external sensing hardware. *(b)*: A touchpad created as a planar conductive area with four connections points, *A, B, C* and *D*.

7.2.1 Knitted Sensor

The knitted sensor design introduced is a continuous conductive rectangular sensing area, constructed using one conductive yarn and four electrode connections points, illustrated in fig. 7.2(b). The yarn used is outwardly conductive, since it is composed of carbon-suffused nylon, and forms a resistive mesh when knitted. The inter-row yarn loop connections form an approximate uniform resistance gradient across the planar surface. This property is beneficial to measuring touch gesture as the resistance gradient is uniform in all directions. Touch location and capacitance magnitude are inferred through voltage measurements recorded at the corner locations. Capacitive touch draws current from the current sources attached at the corners, which affect the differential charge of the circuit (fig. 7.3(a)).

Patterns which have been used in this work so far, developed in previous work, and described in section 2.1.2, as seen in fig. 7.2(a), utilize a single linear conductive pathway that covers a planar area while following the serpentine pathway of weft knitting. In those sensors, location is inferred as a linear distance along the pathway mapped along a 2-dimensional surface. Their circuit requires two connections at the endpoints of the pathway (A and B). This design strategy reduces the dimension of

the measured voltage outputs to what is required to decouple linear touch location and capacitance, and improves the simplicity of connecting the textile to measurement hardware. This strategy could be helpful for inferring static touch location or simple swipe gestures, but it is non-ideal for inferring complex gesture pathways across the surface of the sensor. The measured output response is non-uniform between geometrically adjacent touch-points which complicates precise localization. Additionally, touch-points are spaced apart into a sparse sensitive area to prevent shunting when two or more touch-points are contacted. This spacing induces loss of contact when touch is transferred to another location, thus breaking the gesture's continuity.



(a) (b)

Figure 7.3: Illustrations of the measurement circuit and example recorded gesture data, produced at CFF [3]. *(a)*: Illustration of the measurement circuit and resistor-capacitor network circuit formed during touch. *(b)*: Plot of example measured data showing the changes in gain measurement during onset, gesture, and offset.

7.2.2 Measurement Circuit

The capacitive touch circuit formed by the planar conductive area is modeled as a mesh resistor-capacitor ladder network. Figure 7.3(a) illustrates the circuit diagrams of the resistances formed between the location of touch and the sensing points (A, B, C, D), represented as a star graph. The values of the resistances vary depending on touch location. The value C_t represents the magnitude of capacitance induced by touch, which is used as a pseudo-pressure. The voltage measurement locations (A, B, C, D) have associated parasitic capacitances $(C_{p_A}, C_{p_B}, C_{p_C}, C_{p_D})$ which affect the

voltages at the measurement locations. The excitation waveform passes through current-limiting resistors (R_A, R_B, R_C, R_D) that reduce the current entering and exiting the fabric, which allows voltage measurements to be discerned.

An example gesture and processed measurement are shown in fig. 7.3(b). Figure 7.4 further explains the waveform processing steps. All measured gestures consist of a distinct touch onset, gesture action, and touch offset. During onset, the measured waveform gains decrease due to the increase in touch capacitance. The gain at each measurement point attenuates as a function of touch location, where attenuation increases as touch proximity increases. This phenomenon can be seen during the gesture action, where the motion trajectory and gain attenuation pf each sensing point correlate. Once the gesture is complete, the offset action returns the gain measurements to a baseline.

Waveform generation and acquisition are performed using a Keysight 33622A arbitrary waveform generator and a Keysight MSO-X-3024T oscilloscope. The waveform generator produces a 2 MHz sine wave used as input to the circuit (output 1) and 250 Hz square wave used to trigger oscilloscope sampling (output 2). The sine wave passes through current-limiting resistors attached to the four corners of the fabric circuit. The resistance values are approximately equal to the touchpad resistance of 4 kOhm/sq. A parasitic capacitance of approximately 60 pF is observed at each measurement point. Gesture data is collected over the span of 1 second at a rate of 250 frames per second. Each frame window consists of approximately 4000 samples collected at 625 MSamples/s. The frames are processed using Bode analysis to return the magnitude ratio (gain) of the input and measured waveforms. The output data is of dimension 250×4 corresponding to the 250 measurement frames and 4 measurement channels.

7.2.3 Signal Filtering

After the data is captured by the four electrodes as changes of gain values over time, a baseline event is subtracted, captured while no gesture was being performed on the knitted touchpad. The purpose of baseline subtraction is to remove any elements of the representation that stay the same across different measurements, therefore highlighting the differences between gestures, which makes

classification easier.

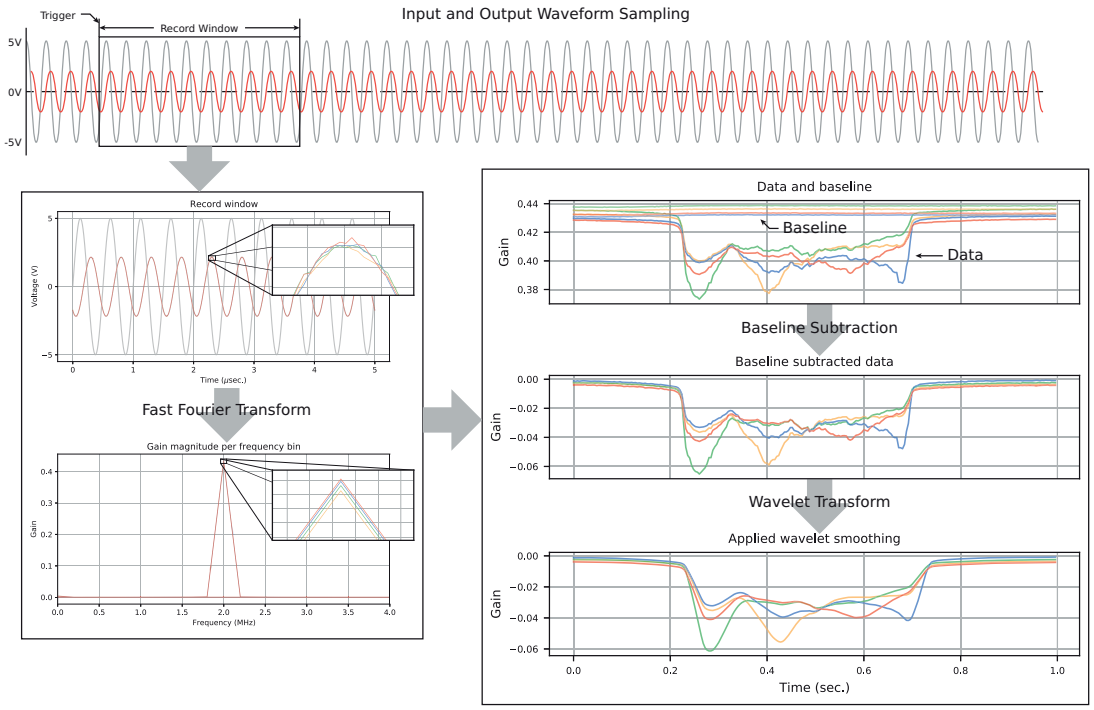


Figure 7.4: Diagram of the data processing pipeline, produced at CFF [3]: The waveform data is sampled in segmented record windows over the duration of the gesture. The FFT of the window is processed and the magnitude of the bin at the input frequency is used to determine the window gain. The measured baseline gain is subtracted from the gesture gain data and a wavelet transform is applied to smooth high-frequency changes.

Subsequently, in order to further reduce noise from the signal, I filter it using *wavelet transform*, a powerful and flexible analytic tool which allows us to obtain signal information from both its time and frequency domains. It does not only indicate which frequencies are present in the signal, like Fourier Transform, but also when those frequencies occur in time. Wavelet Transform achieves this by calculated compromises: by low frequencies having a high resolution in the frequency domain, and low resolution in the time domain; and high frequencies having a low resolution in the frequency domain, and high resolution in the time domain. Wavelet Transform analyses the signal in different scales: first, working with larger windows of the signal for elements stretched in time, like low frequencies. Then, after that information is acquired, we can progressively make the window of interest smaller to look for information in those scales. As we shrink the window, we lose the ability to capture low frequencies, however, that information should have been obtained in the previous

step. Ultimately, we are able to use the collected information from different scales to reconstruct the signal.

In order to filter the signal, the signal is first deconstructed through a *Symlet* mother wavelet with 4 vanishing moments. After having obtained all the elements necessary for reconstruction through analysis at different levels, the signal is reconstructed using only a subset of them. Typically, the higher frequencies correspond to noise, so they are removed. For this analysis, only the first 4 levels of components are retained. These settings were selected such that noise is removed, but the filtered signal follows the original closely. Figure 7.4 illustrates an example of the original signal after the baseline has been subtracted, as explained above, together with its filtered version through wavelet reconstruction.

7.2.4 Neural Network Architecture

The neural network architecture used to build the recognition model relies on a combination of CNNs and LSTMs. Deep learning is chosen over traditional machine learning techniques based on the strengths that both CNNs and LSTMs have shown in representing time-series signal data. Additionally, these networks automatically capture the important aspects of the data without needing feature construction, typically required by most non-neural network algorithms to represent a time-series signal.

The neural network takes as an input the data after being processed as described above. The architecture, illustrated in fig. 7.5, starts with a one-dimensional convolution layer with a rectified linear unit (ReLU) activation function, followed by a batch normalization layer and a dropout layer for regularization purposes. It continues with another one-dimensional convolution layer, again with an ReLU activation function, followed by a max-pooling layer. The function of convolution is capturing local features at different scales in the signal. Subsequently, three layers of LSTM cells follow, each with a hidden size of 100. LSTMs have been widely used for sequential data analysis, including time-series signal data, since they capture the signal's temporal dependencies. Among recurrent layer variants, they are chosen since they are capable of handling long-term dependencies in sequential data [50, 126].

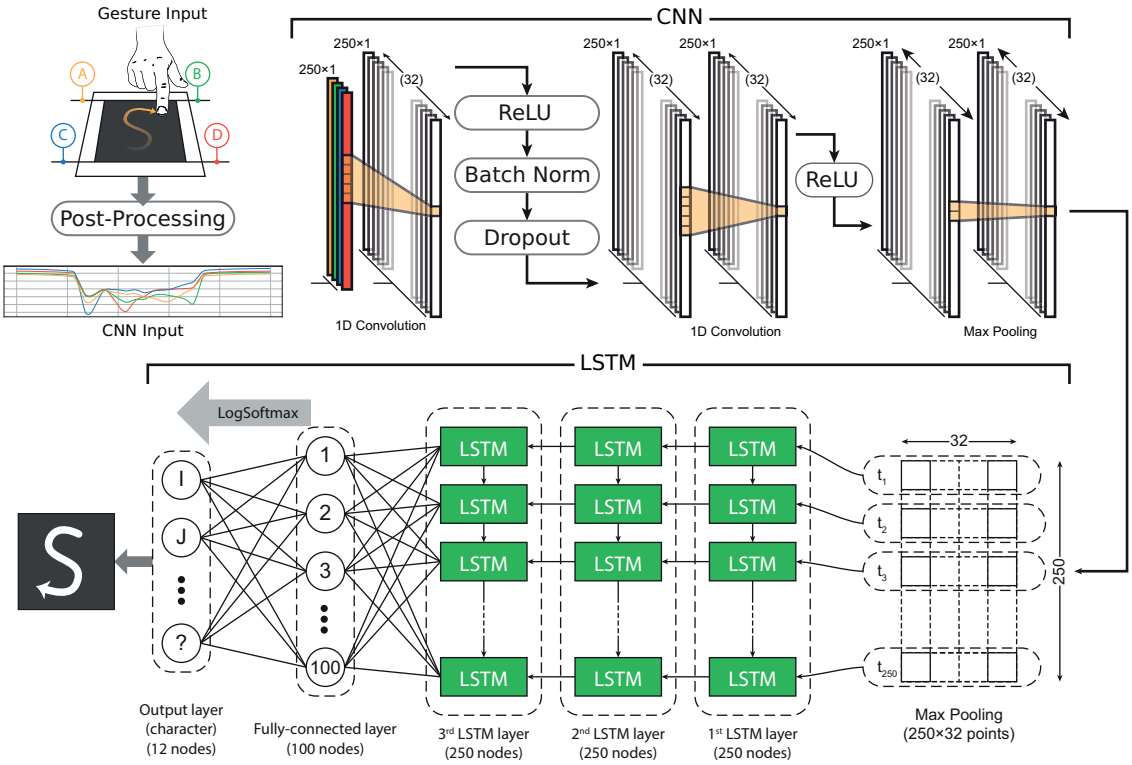


Figure 7.5: CNN-LSTM neural network architecture: the four-signal representation of a gesture event is used as an input after filtering, and the output is a predicted gesture category. The CNN component captures spatial relationships in the signal pattern, while the LSTM component models time-dependency. Figure produced in collaboration with CFF [3].

Each input to the system consists of one gesture event as captured by the four electrodes connected to the sensor, and after being transformed to filtered frequency gain values. The sequence is 250 time steps long, and there are 4 values per time step, corresponding to an input layer of size 4 to the neural network. The temporally-related frequency gain values are transformed to one of 12 possible gesture classes through a 12-output linear layer at the end of the network. In order for the network to predict a correct gesture, it needs to be trained with user data, an aspect discussed in section 7.3 and section 7.4, together with the training hyper-parameter choices.

7.2.5 Model Deployment

For an interactive system, after the above architecture is used for training, the resulting model needs to be deployed to lightweight hardware. A NVIDIA® Jetson Xavier™ NX Developer Kit (fig. 7.6), running Ubuntu 18.04 with 8GB of RAM, a 6-core NVIDIA Carmel ARM CPU, and a NVIDIA

Volta GPU, is used for gesture classification (table 7.1). The NVIDIA board's ability to do fast parallel math is important, as it allows running pre-trained models. In addition, its form factor is relatively compact, an important feature for many of the potential applications of the sensor.

Currently, the data used as the model input is a *.csv* file containing one user-captured touchpad gesture. First, the baseline readings are subtracted and waveform filtering is performed to prepare the data for evaluation. Subsequently, the resulting waveform is input to the model for classification. The model then returns its prediction as an output, which is one of the 12 gesture classes.

In the future, the model deployment should be adapted to handle continuous or real-time signal classification, as opposed to only pre-recorded discrete signals. A barrier to real-time signal classification is the practical lack of hardware that can perform the signal generation, acquisition, and filtering at the speeds necessary to continuously supply the system with new inputs. Custom hardware would be able to stream the acquired signal to the NVIDIA Jetson board for classification. Once the Jetson board would acquire the signal, it would need to parse it to isolate individual gesture windows, since the model would have been trained on inputs containing only a single gesture. An additional model may be necessary to accurately distinguish gestures. The main gesture classification model could then take gestures as inputs as they are returned from the parser.

Figure 7.6: NVIDIA Jetson Xavier NX Developer Kit [4]

Table 7.1: Hardware Specifications

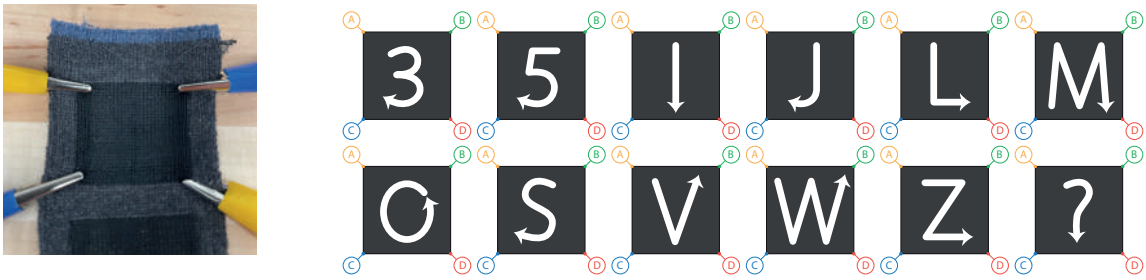
GPU	NVIDIA Volta Architecture with 384 CUDA cores and 48 Tensor cores
CPU	6-core NVIDIA Carmel ARM Processor
RAM	8GB LPDDR4x
Size	70 mm x 45 mm

7.3 Experiments

In order to train the neural network models, user data is required, which was collected using the circuit described in section 7.2.2. A touchpad sensor like the one shown in fig. 7.7(a) was used, connected to a Keysight 66322A Series Waveform Generator and a Keysight InfiniiVision MSO-X

3024T mixed signal Oscilloscope, for signal generation and acquisition respectively. A group of 8 total college-age subjects participated in data collection. All participants were in good health. For a single trial, participants drew one of the 12 gestures included in the study onto the touchpad sensor, as the oscilloscope was capturing the resulting waveform. The gesture pathways captured are illustrated in fig. 7.7(b). For each gesture, a baseline reading was also captured without any input to the touchpad, to be further used for data processing as described in section 7.2.3. A few of the subjects completed trials over different sittings. In those cases, a baseline measurement was taken for each sitting to account for outside conditions.

Training was accomplished using leave-one-out cross validation for each of 5 subjects. Additionally, I evaluated the performance of the trained models, by testing their accuracies against data from 3 new subjects, to further investigate the robustness and reliability of the results. Each participant performed at least 20 trials for each of the 12 gestures included in the experiment, with an average of 45 samples. A total of 2700 samples were collected for cross-validation, with an average of 225 samples per class. The evaluation set was composed of a total of 720 samples, or 60 per class, with each subject performing 20 trials per gesture type. I investigate whether the gesture recognition model is capable of accurately distinguishing among 12 different language character gestures. The collected and processed data is used as an input into the CNN-LSTM gesture recognition network, which outputs one of the 12 possible gestures. Accuracy, precision, recall, and F1-score are used to determine its performance. More details about the process of training and evaluation are provided in the section below.



(a) (b)

Figure 7.7: Pathways of the collected gestures. *(a):* The knitted component of the sensor on which the gestures were performed; *(b):* All the gesture pathways that were collected and analysed [3].

7.4 Methods and Results

Data from 5 subjects was used for training and cross-validation, while data from 3 other subjects was used to evaluate the trained models. During cross-validation, at any time, data from 4 subjects was used for training, and data from the one other subject was used for validation. Therefore, 5 total trained models were produced, each of which was evaluated using evaluation data. Any one of those models could be deployed for use in a real-time interactive system.

The subject data was acquired as described in section 7.3, and processed as detailed in section 7.2.3. Before training, the data was balanced by oversampling within each subject, such that each class had the same number of samples, helping to achieve model stability. The balanced training dataset resulted in an average of 277 samples per class, while the evaluation dataset was already balanced at 60 samples per class. The model, whose architecture was defined in section 7.2.4, was trained using PyTorch 1.4 [104] with CUDA 10.0, using a learning rate $\alpha = 0.001$, a dropout rate = 0.6 and a batch size = 128, over 2000 epochs. *Adam* was used as an optimizer, and a negative log likelihood loss function. The network weights were initialized using Xavier initialization [43].

These hyper-parameters were the same as the ones in section 5.3, where an LSTM-based neural network model was used to identify location of touch on a 36-button knitted sensor. In addition to using the subject data to train our CNN-LSTM gesture recognition model, I also use it to train an LSTM-based model as a baseline comparison. The only difference between the two methods is the neural network architecture. The technique used in section 5.2 also involves a statistical feature construction step over the 92 analyzed frequency gains, which was not possible to apply to the current data, since, due to hardware constraints, only relies on a single frequency. The results of using the baseline architecture are included in table 7.2, together with those of the neural network model.

7.4.1 Cross-Validation and Evaluation Results

The model's cross-validation *accuracy*, reported in table 7.2, is 78.6%. This accuracy was calculated as the average accuracy of the 5 cross-validation folds. This means that the data from one subject at

a time was tested against the model trained on the data from the other 4 subjects, and subsequently, the average of the results is reported. The accuracy for each fold was calculated as the average of the last 50 epochs. The test dataset performance, which aims to capture how generalizable the model is, shows an even greater accuracy of 88.5%. This accuracy was calculated as the average of testing the 3 test subject data against the 5 trained models produced during cross-validation.

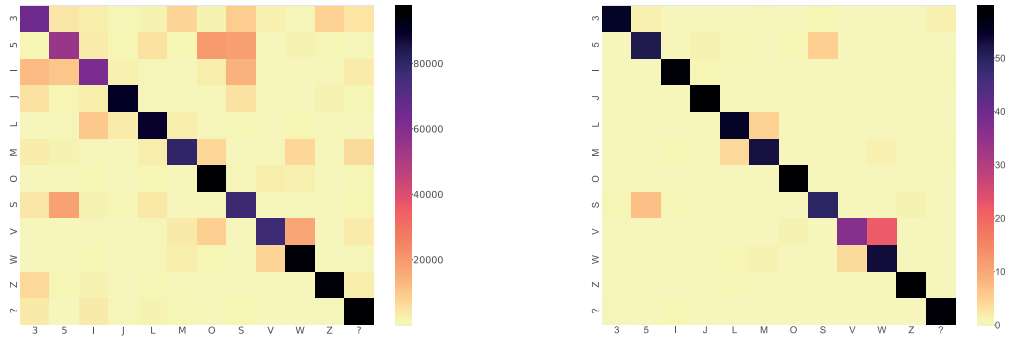
Table 7.2: Classification results for model cross-validation and evaluation.

	Cross-Validation				*Evaluation*			
Method	*Accuracy*	*F1-Score*	*Precision*	*Recall*	*Accuracy*	*F1-Score*	*Precision*	*Recall*
LSTM	41.7%	22.4%	25.4%	22.0%	52.5%	53.9%	61.6%	52.5%
CNN + LSTM	78.6%	73.5%	73.9%	73.7%	89.8%	89.7%	90.3%	89.8%

Other measures of performance calculated are *macro-precision, macro-recall*, and *macro-F1-score*. Precision refers to the ratio of true positives to the combined number of true and false positives. Recall is the ratio of true positives to the combined true positives and false negatives. The F1-score is the harmonic mean of precision and recall: $F1 = 2 * (precision * recall)/(precision + recall)$. Macro-precision, macro-recall, and macro-F1 refer to the balanced respective scores per class, and those values are obtained by averaging all respective class scores. In table 7.2, these values are simply referred to as: *precision*, *recall*, and *F1-score*. These scores give greater insight into how specific accuracies were achieved. Additionally, the classification matrices of all gestures in fig. 7.8 are provided, for both cross-validation and evaluation of the CNN-LSTM model.

As seen from fig. 7.8, there are gestures that incorrectly get classified as others, with pairs such as *'S'* and *'5'*, or *'V'* and *'W'* being noticeable. This can be expected as the trajectories between the two gestures are very similar. It should be noted that these results are not rotation-invariant, due to the nature of the gestures selected. Otherwise, the users' intention regarding entering an M versus a W would be obscured. Overall however, the model accurately identifies gestures performed on it, and generalizes very well, most probably in part, because the architecture includes batch normalization and a relatively high dropout rate of 0.6. If we compare the performance of this model to the baseline, we notice that the CNN-LSTM model significantly out-performs the baseline

in all measures.



(a) (b)

Figure 7.8: Classification matrices produced with results from cross-validation *(a)*, and evaluation *(b)*. The matrix rows denote the true row categories of the gestures, while the columns show the ones predicted during evaluation. Each category is denoted by the respective gesture pathway performed on the knitted touchpad. A higher-value diagonal, with the rest of the matrix having lower values, would be a desirable combination indicating a high accuracy. This figure illustrates how most gestures are correctly classified - if gestures are similar to each-other however, it is easy to incorrectly classify them as each-other.

The accuracies reported here are subject-independent, even though subject-specific calibration is expected to increase overall accuracy. If a knitted pattern is designed to be used by a single user, calibration is likely to increase the gesture detection accuracy, since it removes variations that are caused by different users' physiological states, or specific ways of performing gestures. However, designing a system that does not rely on calibration makes the experience of multiple users interacting with the same sensor much more intuitive and non-intrusive. Additionally, it would be expected that provided more data is available, the model's accuracy would increase, since neural network models typically require a large amount of data to produce accurate results. Moreover, training under a variety of conditions, such as different environmental condition and other possible distortions, would mean that the model would incorporate the signal variations induced, and be more robust when exposed to them.

7.4.2 Resources and Time Performance

The machine on which our models were trained through cross-validation, and later evaluated was running Ubuntu 18.04, with an Intel® Core™ i9-9960X CPU @ 3.10GHz, 128 GB RAM, and 2x

RTX 2080 Ti Blowers with NVLink cards. The trained models were deployed and evaluated on a NVIDIA Jetson Xavier NX Development Kit, described in section 7.2.5, which was also running Ubuntu 18.04. The total run-time of 3 different stages of the evaluation process was measured: the time to convert the input from a NumPy [54] array to a CUDA tensor, the time taken for the loaded model to evaluate the input tensor, and the total time taken for one trial including the first two measurements. The results of these measurements are included in table 7.3.

Table 7.3: Timing measurements of model execution

	Conversion to CUDA	*Sample Evaluation*	*Total Trial*
Bootstrap and First Trial Time	3.7s	3.6s	7.3s
Avg. Trial Time after first	1.9×10^{-4}s	1.7×10^{-2}s	3.0×10^{-2}s

The purpose of this evaluation was to determine the feasibility of building a responsive real-time interactive system given a trained gesture recognition model. From the results in table 7.3, we can see that the average response time of a typical evaluation is approximately 30 ms, a very reasonable response time for real-time systems. The first trial's response time is much greater since it involves bootstrap and the NVIDIA CUDA® Deep Neural Network library (cuDNN) performing cache allocations. Typically, that first trial in real-world applications can be substituted with a sample that is not relevant to the gesture signal evaluation. For a fully-interactive system, the response time of the signal acquisition and processing controller would need to be added to that response time. Even though that component is not implemented in this work, chapter 5 of this work has shown that it is possible to have a responsive micro-controller performing those functionalities. Such systems could enable many applications, some of which are discussed in the following section, together with other considerations for real-world interactivity.

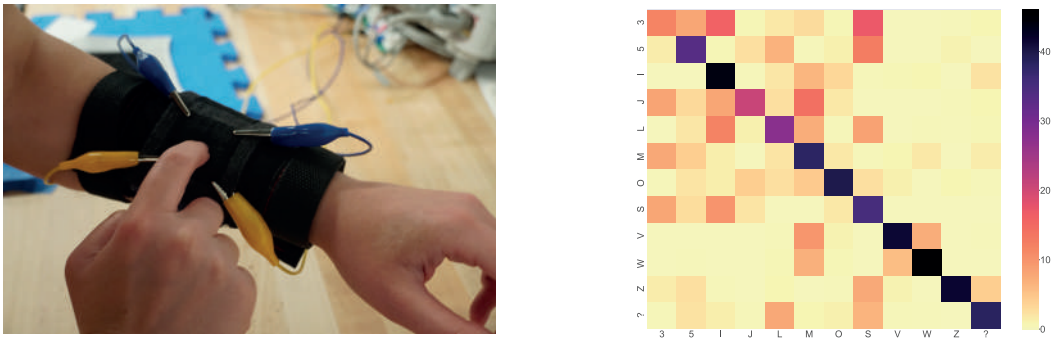
7.5 Real-World Applicability

The experimental results in section 7.4 are encouraging for the feasibility of our proposed technology, however they were performed in a controlled laboratory setting. Specifically, for all those experiments the sensor pinned in a static position on the table during data acquisition. In order to expand the experimental set up of CTS gesture pads and evaluate their functionality and accurately in

a setup resembling the real world, there are many other aspects to be considered. This section begins discussing and investigating some of these possibilities, even though further explorations are necessary. First, a new user study is conducted, with subjects wearing the sensor while inputting gesture data. Next, the effect that washing and drying have on the resistance of the sensor is evaluated. Finally, the application potential and integration of this system are discussed.

7.5.1 Model Performance While the Sensor is Being Worn

One of the main ways the proposed sensor is expected to be used in the real world is via clothing integration, which is facilitated by the compact sensing area. The sensor in proximity to the human body induces capacitance, which is the principle upon which its circuit design relies. When worn near the skin, there is an additional capacitance induced, greater than the baseline parasitic capacitance. Ideally, when worn, the skin should not contact the sensor. Shielding the sensor from underside contact is currently achieved by knitting a back layer on the sensor and routing conductive yarn only on the top layer.



(a) (b)

Figure 7.9: User study to test model performance while knitted sensor is being worn. The setup in *(a)* shows the knitted sensor fixed on a removable Velcro-strapped pad to be worn on the forearm [3]. Four electrodes are connected to the corners of the sensing area, similarly to fig. 7.7(a) for data collection. The heatmap generated in *(b)* shows the classification results for each gesture pathway. The matrix rows denote the true row categories of the gestures, while the columns show the ones predicted during evaluation. There is a clear difference between this figure and the heatmaps in fig. 7.8. However, the diagonal in the middle is still distinguishable from the rest of the values.

Methods and Results

In order to further examine the robustness of the gesture recognizing model introduced above and its usefulness in the real world, a new user study with 3 subjects was conducted. The study is similar in design and analysis to the cross-validation and evaluation studies described in section 7.3 and section 7.4, but in this case, the subjects were wearing the sensor on their forearm while performing the same 12 gestures as in the studies above. The sensor was pinned on another Velcro-strapped pad as shown in fig. 7.9(a). Subjects were not moving while collecting the data, but some motion of their arms would be expected. Each subject performed each of the 12 gestures 20 times, with a total of 720 samples, or 60 per class collected. This dataset was tested against the already trained models from the cross-validation study. No additional training was performed to account for this new condition.

Table 7.4: Classification results for gestures performed while the sensor was being worn.

	Accuracy	*F1-Score*	*Precision*	*Recall*
CNN + LSTM	58.0%	58.2%	61.2%	58.0%

Table 7.4 demonstrates the results computed for the same performance measures as the studies above, including: *accuracy*, *precision*, *recall*, and *F1-score*. Figure 7.9(b) illustrates the classification matrix of the real and predicted gestures. The computed accuracy is 58.0%, which is lower than both the cross-validation and testing accuracy, however well above chance accuracy (8.3%). Increased accuracy would be expected with training that includes samples collected while the sensor is being worn. For a complete system implementation, training should be performed under a large variety of conditions to ensure robustness to different real-world circumstances.

7.5.2 Effect of Washing and Drying on Sensor Resistance

Another aspect in evaluating the robustness for use of a knitted sensor is the effect of washing and drying on its conductivity. The resistance of the sensor is an important property in representing the resulting signal, and subsequently building a gesture-recognizing learning model. Minor changes

in resistance from activities like folding and stretching are anticipated and can be accounted for within the sensing and signal processing pipeline. Furthermore, laundering is an essential post-processing step in the manufacturing process that permanently sets physical yarn properties, such as expanding heat-bulking Nylon fibers, which in turn alter the baseline electrical conductivity. The sample tested has been washed and dried before these experiments, in addition to being steamed, as part of its manufacturing process. In this experiment, first, the baseline resistance across every pair of connection points illustrated in fig. 7.10 is measured. Then, the sensor is washed and dried for five cycles, and the resistance is measured across the same pairs of points after each cycle.

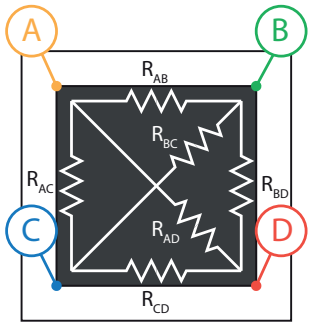


Figure 7.10: Annotated sketch of knitted sensor, showing points along which resistance was measured [3].

Planar conductivity involving multiple connection points is described using the symmetric matrix shown in (section 7.5.2), which is an extrapolation of *Kirchhoff's Current Law* stating that the sum of the currents entering a node is equivalent to the sum of the currents exiting it. In this application, conductivity, G, is directly proportional to current, I, and inversely proportional to resistance, R, such that $I \simeq G = R^{-1}$. The inverse of the resistance values indicated in fig. 7.10 comprise the non-diagonal elements of the conductivity matrix. The change in conductivity is assumed to be scalar in that the values of the conductivity matrix will change proportionally. In practice, these values vary due to local changes in conductivity. Therefore, the average change in values is used to describe the cumulative change in conductivity as the matrix G_{ABCD}:

$$\begin{bmatrix} -(G_{AB}+G_{AC}+G_{AD}) & G_{AB} & G_{AC} & G_{AD} \\ G_{AB} & -(G_{AB}+G_{BC}+G_{BD}) & G_{BC} & G_{BD} \\ G_{AC} & G_{BC} & -(G_{BC}+G_{AC}+G_{CD}) & G_{CD} \\ G_{AD} & G_{BD} & G_{CD} & -(G_{AD}+G_{BD}+G_{CD}) \end{bmatrix}$$

Procedure and Results

The sensor was washed according to the American Association of Textile Chemists and Colorists (AATCC) Laboratory Procedure 1-2018 Home Laundering: Machine Washing protocol [1]. This protocol specifies a

35–minute wash duration with 1.8 kg of laundry and 66 ± 1 g of detergent, and a standard tumble drying protocol with a temperature of $68 \pm 6°C$. This protocol was chosen as appropriate for everyday laundering of clothing.

For the proposed sensor, 6 resistance values were measured for each of the two conditions: *baseline (b)*, and *washing and drying (d_n)*, where $n = 1$ to 5 indicates the cycle number. The values were measured between every pair of corners in the sensor, annotated as A, B, C and D in fig. 7.10, which represent the connection points to the measurement hardware. To fully characterize the resistance across the conductive area of the sensor the resistance between each pair of connection points is necessary. For each resistance measurement, 100 samples were captured using a Keysight 34465A digital multi-meter, which were then averaged to represent the value of that measurement. The results are included in table 7.5. For each test, the percent change in resistance between the baseline measurements and measurements after each washing and drying cycle was calculated as $\%\Delta R_{(b,d_n)} = (R_{d_n} - R_b)/R_b$. In this case, R_{d_n} stands for the resistance value between the two points after the n^{-th} washing and drying cycle (d_n), and R_b for the baseline resistance value between those same points, before any of the washing and drying cycles recorded. The cumulative change in resistance is calculated from the cumulative average of the element-wise matrix division of the baseline and drying cycle conductivity matrices formed using the relation in (section 7.5.2), where $\%\Delta R_{(b,d_n)} = avg\left(G_b / \left(G_{d_n} - G_b\right)\right)$.

Table 7.5: The percent change in resistance ($\%\Delta R$) between the *baseline (b)* and each of the experiments after *washing and drying (d)* is reported. The resistance values are measured between all pairs of the sensor's connection points. The cumulative change in resistance is also reported.

	$[A,B]$	$[A,C]$	$[A,D]$	$[B,C]$	$[B,D]$	$[C,D]$	*Cumulative*
$\%\Delta R_{(b,d1)}$	−6.12%	28.09%	8.52%	12.32%	8.93%	1.77%	8.10%
$\%\Delta R_{(b,d2)}$	7.92%	31.48%	29.84%	31.26%	20.52%	15.44%	22.16%
$\%\Delta R_{(b,d3)}$	0.77%	8.82%	14.08%	18.42%	17.98%	0.55%	9.69%
$\%\Delta R_{(b,d4)}$	7.05%	15.36%	17.09%	20.18%	13.20%	−2.65%	11.23%
$\%\Delta R_{(b,d5)}$	−2.10%	11.07%	7.42%	11.03%	−3.80%	−2.21%	3.19%

The results in table 7.5 show that there is stability in the resistance measurements of the sensor after washing and drying it. However, there is a change in the cumulative resistance across washing and drying trials. The fact that the change in overall resistance does not substantially increase from trial to trial is promising. However, the change in resistance varies from approximately from 1% to 31% for individual connection point pairs. In addition, values in the second washing and drying trial d_2 seem higher compared to previous and subsequent trials, possibly due to a measurement error. Further testing is necessary to

investigate the effects of washing and drying more comprehensively, and this observed variability needs to be incorporated into the model design, so that gestures recognition is stable for any applications depending on it.

7.5.3 Building Interactive Applications with Gesture-Recognizing Knitted Sensors

The system components described in section 7.2 create the foundation for building an interactive system that uses a knitted sensing area to accept gesture input and a machine learning model to determine the gesture performed by the user. In a real-time system, gestures can subsequently be interpreted by the specific application to trigger different events. This technology enables many kinds of applications, and additionally offers extensibility. Figure 7.11 shows two related views of the system. On the left, it illustrates how data can be collected from users to train and evaluate a machine learning model. Gestures can be determined by the application needs, and the experiments and results in this work show that it is possible to recognize even relatively complex gestures, such as letters and numbers, with high accuracy. This allows great flexibility in the choice of gesture sets for training, even on a small-sized sensing area. Training happens offline on a server and after the cross-validation results achieve the required accuracy, the trained model is further evaluated, and then deployed on a smaller system, such as the NVIDIA Jetson computer.

On the right side of fig. 7.11, the implementation of an interactive system based on this technology is illustrated. A user enters a gesture input on the knitted sensing area, connected to an embedded system for signal acquisition and pre-processing. This system would communicate in real time with the NVIDIA Jetson board hosting the trained gesture recognition model, capable of interpreting the signal as the gesture the user intended. The application relying on this technology would then respond to the user based on the meaning assigned to the particular gesture.

This system configuration is extensible in the types of gestures recognized, since the model can be re-trained offline and re-deployed on existing hardware, allowing for large-scale production. The developments introduced in this work hold promise for creating innovative applications. Some potential examples are broadly described below:

- *Character Recognition:* The gesture examples used for recognition in this work are a subset of the character set of the English language. They were used to explore the feasibility of constructing a character recognition system. Having a system trained on the whole character set would allow written

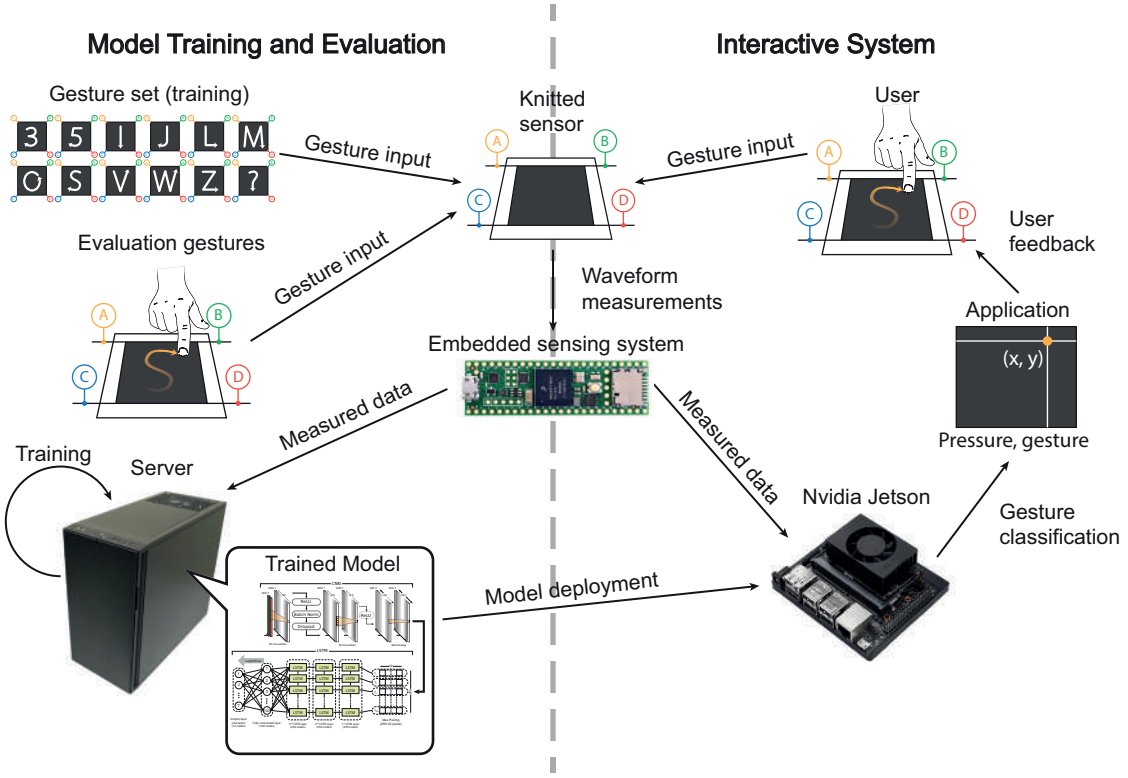


Figure 7.11: Processes and component interactions that describe the model creation and the working of an interactive gesture recognition system. Data collection, training and evaluation happen off-line and typically require more time and computing power. Once a model is trained to high accuracy, it can be deployed on lightweight hardware to recognize gestures in real time, supporting different interactive applications. This diagram was produced in collaboration with CFF [3].

messages captured through fabric sensors to be transmitted. Since letters already have meaning embedded in them by convention, such an application would be intuitive, easy-to-use, and have great expressive power.

- *Controllers:* Another category of applications that can be built using this sensor is that of controllers. This sensor allows the emulation of existing controller functionality, but with the added flexibility of fabric. For example, users can control their phone functionalities such as accepting or rejecting a phone call, changing the music, and more, through knitted interactive areas in their clothes, capable of recognizing gestures. Home automation is another potential application area, with such sensors integrated into furniture, pillows, or blankets, giving controllers a softer, more tactile-friendly quality. Gaming could benefit from application where knitted sensors are used as portable, foldable, and lightweight alternatives to hard-electronic controllers. Additionally, pressure sensitivity, an aspect of these sensors only explored in a limited way so far [133, 132, 93], could offer new interactive modalities

for gaming and other applications.

- *Gestures in 3D Space:* Gestures do not need to be confined into a 2D plane. Knitted fabric can flex, fold, stretch, and move dynamically. Instead of only considering those fundamental aspects of fabric behaviour as qualities to design out, in order to maintain stability of touch or gesture representation, for certain applications we can also choose to design with these qualities at the center. For example, stretching could be given a specific meaning in a interactive system, and so can folding the fabric, twisting, or pinching it. An important aspect that needs to be considered while designing for such use cases however, is that capacitive sensing, the sensing strategy on which these sensors are designed, requires the presence of two conductors in proximity. In the typical cases, also explored in this work, the two conductors are the conductive yarn and human skin.
- *Virtual and Augmented Reality (VR/AR) Applications:* Knitted sensors with gesture recognition technology integrated into them can be useful for VR/AR environments. It is easy to imagine objects that can be designed somewhat generically using knitted fabric with interactive gesture-sensing areas, with their functionality depending on the specific VR/AR application. This sensor's construction process allows for scalable design, resulting in interactive shapes capable of being built in different sizes. Therefore a whole environment could be composed of soft, knitted interactive objects, which take a different meaning and functionality, depending on the application and visuals overlayed on them. Additionally, such applications could be especially useful for kids, since they offer more safety than potential hard-electronic equivalents.

7.6 Conclusion

This chapter creates the foundations for building an interactive gesture recognition system using an easily manufactured capacitive weft-knitted sensor, and a classification model that recognizes 12 English language characters as gestures. Additionally, the model is deployed on NVIDIA® Jetson Xavier™ NX, a small, lightweight, and powerful embedded system-on-module.

The sensor introduced in this work uses a single carbon-coated conductive yarn to create a continuous sensing area. That yarn is combined in the knitting process with other regular yarns to produce the final fabric substrate, similarly to other sensors of varying designs produced using this technology. Four electrodes are attached to the corners of the rectangular conductive area to measure voltage across it. Different gestures produce different voltage responses from the four connection points.

In order to capture those identifying differences, I train and evaluate a CNN-LSTM neural network model with data from a total of 8 subjects. Leave-one-out cross-validation was used for training on 5 subjects, achieving a subject-independent accuracy of 78.6%, and the data from the 3 other subjects was used for further evaluation of the model's generizability, during which, the model achieved a subject-independent accuracy of 89.8%. The response time of the trained model was also tested, after the model had been deployed on the Jetson system. The average response time was 30 ms, which is very encouraging to building a real-time application. These technical contributions help us advance toward feasible interactive embedded systems, capable of recognizing gestures on knitted senors.

Additionally, another user study explored the model's accuracy response with gesture data collected while the sensor was being worn. A subsequent experiment consisted of measuring the sensor's resistance across all connection points, after washing and drying it. These considerations, together with the discussion of the components and processes necessary to build real-time applications upon this technology, help advance the capabilities of knitted textiles toward their use in real-world environments. Moreover, the diverse contributions presented in this dissertation pave the way for future advancements in several areas, some of which are examined in the next chapter.

Chapter 8: Summary & Future Directions

The prospect of human-centered designs relying on knitted capacitive touch sensors is fascinating, and its feasibility is illustrated throughout this work. The advancements introduced here hold promise for building many interactive application which rely on touch-sensitive fabric as an input medium, even for complex gestures. This work synthesizes information from a variety of fields, such as machine learning, human-computer interaction, algorithm design, software design, signal processing, and system design. The section below summarizes its main contributions, followed by open areas of investigation to further the capabilities and reach of technology.

8.1 Contributions Summary

I started investigating the feasibility of building an interactive system based on knitted sensors by first extracting the signal response of human touch on the existing knitted sensor [133]. I designed a 13-subject user study, during which each subject was asked to press on each of the 36 touch locations on the knitted sensor several times, and the sensor output signals were recorded for each location. The results from this study determined it was possible to extract invariant features from the coupled signals output by the knitted sensor system, information which could help identify location of touch. To that end, I designed *Mixed-Source Description (MSD)*, an algorithm to detect invariances in the scale-space of multi-channel signals, and represent them in terms of their most important aspects. Additionally, I introduced *Euclidean Levenshtein Distance (ELD)*, a generalizable distance metric, which measures the similarity between two tensors of varying lengths. Its purpose was to compare the similarity of two key-press events on the sensor, regarding location of touch, so it was possible to evaluate the effectiveness of *MSD*. During this work, the signal received from the knitted circuit was processed using Bode analysis, which calculates frequency gains for a single frequency over each time step. This process is described in detail in chapter 4 [132]. While a touch location recognition model had not yet been constructed, these results indicated the viability of building such a system.

The next component of my work involved building the touch location recognition system whose feasibility was shown, as described above, as well as exposing it to some real-world conditions. I investigated precise

touch location identification on the single-yarn knitted sensor circuit without subject-specific calibration. Obviating the need for calibration makes the sensor more intuitive and inviting to use in the real world, even though calibration facilitates obtaining higher accuracy in machine learning models, due to the similarity of data collected from one subject. I proposed a *recognition model* which relies on a statistical, frequency-based feature construction component, and a long short-term memory (LSTM) neural network classifier, addressing the challenges of minimal information output from the single-yarn system, as well as noise in the signal. In this model, Bode analysis is used as well, but over multiple frequencies, instead of only one. In order to train and evaluate the recognition model, I designed and analyzed *a series of user studies* to move towards real-time, real-world knitted sensors, achieving favorable results, even though some of the studies were limited in size. These studies were designed to enable the creation of a robust location sensing model; demonstrate its generalizability with new users; and investigate its robustness under some real-world conditions. One of the benefits of using digital weft-knitting to construct the fabric sensors on which I focus is the ability to create different patterns of interactive areas on the 2-dimensional sensor surface. The recognition infrastructure I proposed, has the potential to be used with other knitted design forms relying on one yarn and two connections, provided each one uses its own individually trained location identification model. This process furthers the flexibility offered by the manufacturing method to extend to computational models as well, enabling a modular end-to-end interactive knitted system framework.

Identifying location of touch on knitted sensors is an important achievement in making knitted sensors viable for real-world use. However, for these sensors to be intuitively integrated into everyday environments, the end user perspective is also necessary. The next part of my work focused on understanding user perceptions and the potential of this knitted sensor technology to enable a variety of interactions. To accomplish that, I designed, ran, and analyzed *a formative study* structured as focus groups, during which users were presented and interacted with several design forms of knitted sensors. Discussions included general thoughts about interacting with such sensors, potential applications of interest, as well as areas of concern and suggestions for improvements. Subsequently, I synthesized that information to offer guidelines to application and interaction designers using minimalistic knitted sensors as an input medium. To further ground this work in the real world and address some concerns raised in the focus group study, I ran *two experiments* corresponding to everyday use cases: stretching, and washing and drying of some sensor samples. A significant next step, capable of unfolding countless opportunities for applications, is gesture recognition, a sentiment also expressed in the formative study. Through another user study, I *started investigating simple swipe gestures*,

as well as touch samples emulating accidental touch events, likely to happen in the real world. After computing the *ELD* between every gesture sample pair, this experiment demonstrated that samples belonging to the same gesture are more similar to each-other than samples of different gesture types. Additionally, samples emulating accidental touch events were also statistically different from the other gesture classes.

After these encouraging results, I explored gesture recognition more deeply, using relatively complex gestures. The following part of my work builds the foundations for later creating an interactive *gesture recognition system*. First, my colleagues and I designed *a novel knitted sensor* more suitable for performing gestures. It contains a solid conductive area in the center of non–conductive fabric, and four connection points to external hardware. Similarly to the 36-button touchpad, the interactive area is composed of carbon-coated nylon, and the sensor was produced using digital weft-knitting. Next, I designed *a neural network architecture* to classify 12 relatively complex gestures performed on the sensor, which are a subset of the characters of English language. As part of creating an accurate recognition model, I trained and evaluated the model using data collected from *two user studies*. This model can, in the future, extend to include the full character set of English or any other language. In addition, the viability of building a real-time system was illustrated by loading the trained model on NVIDIA® Jetson Xavier NX, a small, lightweight, and powerful embedded system-on-module. Moreover, this system was further tested for robustness toward real-world conditions: the trained recognition model retained accuracy when the test data was collected while subjects were wearing the sensor, and the resistance of the fabric component showed stability after several cycles of washing and drying.

8.2 Future Directions

The exploration space which this work unfolds consists of a variety of areas and directions, including algorithms, machine learning, real-world effects, signal processing, interactive systems, interaction and application design, and user experience studies. Below, I summarize some aspects that should be investigated in the future, for AI-powered interactive systems that use minimalistic knitted sensors as an input medium to reach their full potential and become ubiquitous technology, seamlessly integrated into our everyday environments, building upon the foundations set in this work.

8.2.1 Recognition Accuracy & Generalizability

Touch localization accuracy should be increased in the recognition models for the touchpad knitted sensor used in the location identification experiments in chapter 4 and chapter 5. As described, several knitted

form factors can be designed through the same manufacturing process and can use the same neural network architecture and signal processing hardware. Currently, however, each recognition model corresponding to a particular form would need to be trained using user data specifically collected by pressing on the buttons defined by that sensor's geometry. Then, the trained model can be used to detect touch in real time. Future work could include further investigation into the shapes that are possible to construct given the constraints of knitting, as well as the optimal designs for high accuracy, taking into account considerations discussed in section 2.1.2. More generalized circuit design and algorithmic solutions could also be explored to capture the touch location independent of yarn routing and design patterns.

Chapter 6 of this work showed that simple swipe gestures on the knitted sensors produced using a single conductive yarn and two external connections could be distinguished from each-other and from emulated accidental touch events. Chapter 7 followed with a more extensive gesture recognition exploration using the same manufacturing strategy, but a four-connection circuit, which allowed more complex gestures to be distinguished. This recognition model showed robustness even while being worn in a lab environment. Gesture recognition accuracy still needs to be increased and more gestures need to be included. Even though each application might require its custom-defined gestures, in order to extend this system, it is necessary to ensure its feasibility and robustness with more subjects and more gesture types. Data collected using different fingers for performing gestures, different finger-placements on the sensing pad, different orientations of gesture trajectory, as well as subjects of different ages will need to be explored, due to possible changes in physiology which affect conductivity. In addition, experiments should be run while the sensor is being worn outside a lab environment, and accidental touch events should be recorded as they happen in the real world. Another area of interest could be exploring the recognition of gestures in 3D space, taking full advantage of the fabric form, capable of stretching, flexing, and folding.

8.2.2 Resistance to Real-World Conditions

The sensor laundering and stretching tests described in section 6.4 and section 7.5.2 start to address some practical questions regarding the sensors' resistance to potential daily use distortions. In section 5.3, this work investigated the model stability for touch location identification under conditions of stretching and exposure to electromagnetic radiation for knitted sensors. However, there are many more factors that need to be explored, such as the sensor exposure to heat sources, or different weather conditions. In addition, the sensors' resistance should be measured while they are wet, while controlling for the level of moisture in the

knitted component. The resistance is expected to change according to the amount of water in the sensor, and future work should investigate this aspect more rigorously. Along those lines, the sensors' resistance should be investigated while wet with electrolyte solutions. Their conductive properties are expected to interfere with the current flow within the fabric circuit, causing the output signal to be different from expected. This use case is especially important for wearable applications since sweat contains electrolytes. Future work should investigate these aspects further and take into account their results when creating models and systems based on this technology. In addition to the resistance, in order to fully test the recognition capability of these sensors, the robustness of the trained models needs to be evaluated under such conditions of possible distortion.

Another aspect to be considered is sensor performance while being worn, integrated into clothing, even though the limited study in section 7.5.1 demonstrated its robustness through encouraging results. More studies are necessary to explore the effect of intense physical activity, especially since such sensors are expected to find applications in athletics. Additionally, sweat, could possibly interfere with conductivity, since it contains electrolytes. If it seeps through the sensor, the overall resistance of the sensor will change, which is expected to affect the quality of acquired gesture data. Moreover, if a user is grounding himself or herself while touching a location on the sensor, his/her conductivity is increased, which will again affect the sensor response. False positives could be induced if a conductor came into contact with the sensing areas of the knitted component. Further studies are needed to fully investigate the extent of the effect of such conditions.

Along these lines, the physical durability of the sensor should also be more closely examined, since such sensors need to be able to withstand exposure to different weather and environmental conditions in everyday life. Experiments are needed to test the ability of the carbon-suffused nylon yarn to resists material aging and abrasion. Methods of surface enhancement and preservation, such as coating and lamination, should be investigated as potential solutions, and their effectiveness needs to be quantified. A thorough evaluation should explore these and other factors, including exposure of the sensors to different environmental conditions and humidity, and the sensors undergoing repeated washing and drying cycles. For real-world use, these factors should not lead to loss of their conductive properties, or their ability to correctly detect touch location on an already trained model.

8.2.3 Novel Interaction Modalities

Other modalities of interaction enabled by the sensor introduced above, such as pressure sensitivity should be studied and implemented. For example, soft touch, which would cause a weak applied capacitance, could potentially pose a problem towards accurate gesture recognition. This limitation should be examined further when evaluating user actions like swipes, which do not exert heavy pressure on the sensor. On the other hand, applications can be developed that use pressure differentiation as a feature. In applications where pressure could be useful, such as gaming for example, soft touch can be interpreted as a less intense action. Pressure sensitivity needs to be properly investigated, as it can be useful in a number of applications, and also mentioned as desirable in the formative study in chapter 6.

Additionally, current sensors do not recognize multi-touch, so only one gesture can be performed at a time on the knitted sensor. From experiments conducted with sensors built using a single conductive yarn and two connections, it is known that if two locations are touched simultaneously, the signals from both touch locations are averaged to resolve a single location in between the two [132]. Future work should involve disambiguation of the number of touch points contacting the sensor, since multi-touch capabilities are necessary for this technology to reach its full potential.

Another area could be experimentation with conductive yarns of different properties. The resistance of the yarn affects conductivity, and future models should also account for that quality.

8.2.4 Interaction Design & User Experience

The qualitative study conducted in this work provided us with some user perspective regarding the many ways touch-sensitive fabrics in general, and these minimalistic scalable knitted fabrics in particular, can be integrated into everyday environments. However, this was a formative study which focused on general ideas instead of a usability study of any particular use case, which could provide a more comprehensive view of its feasibility or adoptability.

The formative study was structured as a focus group, which has many benefits in terms of exploring ideas from several different perspectives, with opinions and perceptions building on those of other participants. However, in such a setting, the speaking time of each participant is relatively low, compared to one-on-one interviews, and in many cases time allocation among participants is disproportionate, since it largely depends on participants taking the initiative to share their ideas. Additionally, individual participant's ideas are typically not explored as in depth or detail as in interviews, due to the faster pace of the process. The

setting is also more public, which might have prevented users from expressing and developing use cases that are more personal.

Another limitation of this study is the fact that the 32 participants were graduate or undergraduate students recruited from one university, a relatively narrow segment of the population. For a more complete investigation, participants of different ages, cultures, and backgrounds should be included in such studies. Our participants also mentioned use cases and applications related to accessibility, however, in order to properly study each of those use cases, participants and researchers that are part of the specific community should be involved in the process.

Usability studies need to be performed to explore the potential for adaptation of knitted sensors into end-users' everyday lives. Some potential applications have been discussed throughout this work. One could also investigate new areas of possible integration for these sensors that are unique to them, and do not rely on existing hard electronic prototypes. Future work could focus on further investigating these aspects of this technology, as well as building specific applications, testing their performance and usability in real-world scenarios, and getting user feedback about design aspects.

8.3 Conclusion

Textiles evolved with humanity, from serving the function to protect and keep warm, to showing personal style, and recently to enhance experience by incorporating increased connectivity into our surroundings. Previous work [133] has shown that sensors relying on minimal hardware can be produced using weft knitting, an industry-standard manufacturing technique, for easy integration into clothes and other textile surfaces. The extensible nature of weft knitting allows the sensor to conform to many shapes, and the simplicity of the structure allows it to be integrated into other textiles. These sensors are typically based on one carbon-coated conductive yarn, which is combined in the knitting process with other regular yarns to produce the fabric. The purpose of this design with minimal wiring is to make manufacturing at-scale easier, by reducing post-processing, and to create user-friendly and robust sensors. Such sensors have the power to become seamlessly integrated into our everyday environments, since they were designed with usability and ease of integration as core principles. However, sensing techniques in [133], which were extensively described in section 2.3.2, only approximate general areas of touch.

This minimalism in wiring and hardware design means that, in order to interpret the signals output from these devices, more complex computational models are needed. Moreover, for integration into real-

world environments, everyday use conditions and user needs and perceptions should also become part of the development process. The contributions of this work, as it is situated at the intersection of artificial intelligence and user experience, have addressed those main areas. First, in the process of enabling high-accuracy touch location identification and gesture recognition, two algorithms, *MSD* and *ELD*, were introduced, and two deep learning models, one based on an LSTM architecture achieving 66% accuracy in classifying 36 closely-related locations of touch, and the other, using a combined CNN-LSTM architecture to classify 12 complex gestures with 90% accuracy. These models were trained and validated with data collected during several user studies, which included data collected under some conditions closer to real-world ones, such as the sensor being exposed to electromagnetic radiation, the sensor being stretched, and data collected while the sensor was being worn. Additionally, the computational models were deployed on portable hardware, creating the foundations for real-time interactive systems. Furthermore, through a formative focus group study, I explored users' views of these fabrics in different contexts, including potential concerns and application areas. Subsequently, steps were taken toward addressing relevant questions. Results from a user study demonstrate that it is possible to distinguish among intentional swipe gestures and accidental contact with the sensor. Additional experiments indicate robustness to the effects of stretching and laundering of the sensors.

The multi-disciplinary nature of this work required considering several perspectives to bringing this technology closer to real-world use. It integrated contributions from fields such as artificial intelligence, algorithms, human-computer interaction, and system design. Now, it has opened up many areas for future research, such as: designing algorithms and deep learning architectures to enable touch location and gesture recognition while relying less on sensor geometry, more thoroughly investigating problematic real-world conditions and potential solutions to them, exploring new interaction modalities through yarns with different properties and multi-touch detection, creating a variety of applications using knitted sensors and conducting usability studies with them. The contributions of this work at the intersection of artificial intelligence and user experience have demonstrated the viability of a future where minimalistically-designed knitted sensors have become harmoniously integrated into our environments, inspiring a variety of novel interactions.

Bibliography

[1] *The American Association of Textile Chemists and Colorists [Online].*

[2] *Atlas.ti [Online].*

[3] *Center for Functional Fabrics [Online].*

[4] *The NVIDIA® Jetson Xavier NX™ Developer Kit.* NVIDIA.

[5] *Otter.ai [Online].*

[6] *SHIMA SEIKI USA [Online].*

[7] Agcayazi, T., McKnight, M., Kausche, H., Ghosh, T., and Bozkurt, A. A finger touch force detection method for textile based capacitive tactile sensor arrays. In *2016 IEEE SENSORS* (October 2016), pp. 1–3.

[8] Agrawal, R., Faloutsos, C., and Swami, A. Efficient similarity search in sequence databases. In *International conference on foundations of data organization and algorithms* (1993), Springer, pp. 69–84.

[9] Aigner, R., Pointner, A., Preindl, T., Parzer, P., and Haller, M. Embroidered resistive pressure sensors: A novel approach for textile interfaces. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (2020), pp. 1–13.

[10] Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., and Farhan, L. Review of deep learning: Concepts, cnn architectures, challenges, applications, future directions. *Journal of big Data 8*, 1 (2021), 1–74.

[11] Anzul, M., Downing, M., Ely, M., and Vinz, R. *On writing qualitative research: Living by words.* Routledge, 2003.

[12] Atalay, A., Atalay, O., Husain, M. D., Fernando, A., and Potluri, P. Piezofilm yarn sensor-integrated knitted fabric for healthcare applications. *Journal of Industrial Textiles 47*, 4 (2017), 505–521.

[13] Atalay, O. Textile-based, interdigital, capacitive, soft-strain sensor for wearable applications. *Materials 11*, 5 (2018), 768.

[14] Babaud, J., Witkin, A. P., Baudin, M., and Duda, R. O. Uniqueness of the gaussian kernel for scale-space filtering. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 1 (1986), 26–33.

[15] Bahlmann, C., and Burkhardt, H. The writer independent online handwriting recognition system frog on hand and cluster generative statistical dynamic time warping. *IEEE Transactions on Pattern Analysis and Machine Intelligence 26*, 3 (2004), 299–310.

[16] Bailly, A., Malinowski, S., Tavenard, R., Chapel, L., and Guyet, T. Dense bag-of-temporal-sift-words for time series classification. In *International Workshop on Advanced Analysis and Learning on Temporal Data* (2015), Springer, pp. 17–30.

[17] Bellman, R., and Kalaba, R. On adaptive control processes. *IRE Transactions on Automatic Control 4*, 2 (1959), 1–9.

[18] Bode, H. W. Relations between attenuation and phase in feedback amplifier design. *The Bell System Technical Journal 19*, 3 (July 1940), 421–454.

[19] Braun, V., and Clarke, V. Using thematic analysis in psychology. *Qualitative research in psychology 3*, 2 (2006), 77–101.

[20] Burkov, A. *The Hundred-page Machine Learning Book*. Andriy Burkov, 2019.

[21] Burns, R. B., Seifi, H., Lee, H., and Kuchenbecker, K. J. Getting in touch with children with autism: Specialist guidelines for a touch-perceiving robot. *Paladyn, Journal of Behavioral Robotics 12*, 1 (2021), 115–135.

[22] Chen, L. *Similarity search over time series and trajectory data*. University of Waterloo, 2005.

[23] Chen, L., and Ng, R. On the marriage of lp-norms and edit distance. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30* (2004), pp. 792–803.

[24] Chen, L., Özsu, M. T., and Oria, V. Robust and fast similarity search for moving object trajectories. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data* (2005), pp. 491–502.

[25] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[26] Ciresan, D. C., Meier, U., Masci, J., Gambardella, L. M., and Schmidhuber, J. Flexible, high performance convolutional neural networks for image classification. In *Twenty-second international joint conference on artificial intelligence* (2011).

[27] Coyette, A., Schimke, S., Vanderdonckt, J., and Vielhauer, C. Trainable sketch recognizer for graphical user interface design. In *IFIP Conference on Human-Computer Interaction* (2007), Springer, pp. 124–135.

[28] Davis, F. The textility of emotion: A study relating computational textile textural expression to emotion. In *Proceedings of the 2015 ACM SIGCHI Conference on Creativity and Cognition* (2015), pp. 23–32.

[29] De Rossi, D., Carpi, F., Lorussi, F., Mazzoldi, A., Scilingo, E. P., and Tognetti, A. Electroactive fabrics for distributed, conformable and interactive systems. In *SENSORS, 2002 IEEE* (June 2002), vol. 2, pp. 1608–1613.

[30] Devendorf, L., Lo, J., Howell, N., Lee, J. L., Gong, N.-W., Karagozler, M. E., Fukuhara, S., Poupyrev, I., Paulos, E., and Ryokai, K. " i don't want to wear a screen" probing perceptions of and possibilities for dynamic displays on clothing. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (2016), pp. 6028–6039.

[31] Dollinger, M., and Vanderlelie, J. Closing the loop: co-designing with students for greater market orientation. *Journal of Marketing for Higher Education 31*, 1 (2021), 41–57.

[32] Efrat, A., Fan, Q., and Venkatasubramanian, S. Curve matching, time warping, and light fields: New algorithms for computing similarity between curves. *Journal of Mathematical Imaging and Vision 27*, 3 (2007), 203–216.

[33] Euachongprasit, W., and Ratanamahatana, C. A. Efficient multimedia time series data retrieval under uniform scaling and normalisation. In *European Conference on Information Retrieval* (2008), Springer, pp. 506–513.

[34] Fabbri, R., Duff, T., Fan, H., Regan, M., de Pinho, D. d. C., Tsigaridas, E., Wampler, C., Hauenstein, J., Kimia, B., Leykin, A., et al. Trifocal relative pose from lines at points and its efficient solution. *arXiv preprint arXiv:1903.09755* (2019).

[35] Fabbri, R., Kimia, B. B., and Giblin, P. J. Camera pose estimation using first-order curve differential geometry. In *European Conference on Computer Vision* (2012), Springer, pp. 231–244.

[36] Faloutsos, C., Ranganathan, M., and Manolopoulos, Y. Fast subsequence matching in time-series databases. *ACM Sigmod Record 23*, 2 (1994), 419–429.

[37] Flitton, G. T., Breckon, T. P., and Bouallagu, N. M. Object recognition using 3d sift in complex ct volumes. In *BMVC* (2010), vol. 1, Citeseer, pp. 1–12.

[38] Fukushima, K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics 36*, 4 (1980), 193–202.

[39] Gaver, W., Beaver, J., and Benford, S. Designing design: Ambiguity as a resource for design proceedings of the conference on human factors in computing systems, april 2003, 2003.

[40] Gemperle, F., Kasabach, C., Stivoric, J., Bauer, M., and Martin, R. Design for wearability. In *digest of papers. Second international symposium on wearable computers (cat. No. 98EX215)* (1998), IEEE, pp. 116–122.

[41] Gers, F. A., Schmidhuber, J., and Cummins, F. Learning to forget: Continual prediction with lstm.

[42] Gilliland, S., Komor, N., Starner, T., and Zeagler, C. The textile interface swatchbook: Creating graphical user interface-like widgets with conductive embroidery. In *Proceedings of the 2010 International Symposium on Wearable Computers* (October 2010), ISWC '10', pp. 1–8.

[43] Glorot, X., and Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics* (2010), pp. 249–256.

[44] Goffman, E. The presentation of self in everyday life. new york: Anchor. 1963 stigma, 1959.

[45] Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. *Deep learning*, vol. 1. MIT press Cambridge, 2016.

[46] Gordon, I., and Lowe, D. G. What and where: 3d object recognition with accurate pose. In *Toward category-level object recognition*. Springer, 2006, pp. 67–82.

[47] Graves, A. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850* (2013).

[48] Graves, A., and Jaitly, N. Towards end-to-end speech recognition with recurrent neural networks. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)* (2014), pp. 1764–1772.

[49] Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., and Schmidhuber, J. A novel connectionist system for unconstrained handwriting recognition. *IEEE transactions on pattern analysis and machine intelligence 31*, 5 (2009), 855–868.

[50] Graves, A., Mohamed, A.-r., and Hinton, G. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing* (2013), Ieee, pp. 6645–6649.

[51] Gu, J., and Jin, X. A simple approximation for dynamic time warping search in large time series database. In *International Conference on Intelligent Data Engineering and Automated Learning* (2006), Springer, pp. 841–848.

[52] Hamdan, N. A.-h., Blum, J. R., Heller, F., Kosuru, R. K., and Borchers, J. Grabbing at an angle: Menu selection for fabric interfaces. In *Proceedings of the 2016 ACM International Symposium on Wearable Computers* (New York, NY, USA, 2016), ISWC '16, ACM, pp. 1–7.

[53] Hamdan, N. A.-h., Voelker, S., and Borchers, J. Sketch&stitch: Interactive embroidery for e-textiles. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2018), CHI '18, ACM, pp. 82:1–82:13.

[54] Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E. Array programming with NumPy. *Nature 585*, 7825 (Sept. 2020), 357–362.

[55] Hasegawa, Y., Shikida, M., Ogura, D., and Sato, K. Novel type of fabric tactile sensor made from artificial hollow fiber. In *2007 IEEE 20th International Conference on Micro Electro Mechanical Systems (MEMS)* (January 2007), pp. 603–606.

[56] Hebdige, D. Subculture: The meaning of style routledge, 1979.

[57] Henze, N., Broll, G., Rukzio, E., Rohs, M., and Zimmermann, A. Mobile interaction with the real world. In *Proceedings of the 10th international conference on Human computer interaction with mobile devices and services* (2008), pp. 563–565.

[58] Hládek, D., Staš, J., Ondáš, S., Juhár, J., and Kovács, L. Learning string distance with smoothing for ocr spelling correction. *Multimedia Tools and Applications 76*, 22 (2017), 24549–24567.

[59] Hochreiter, S., and Schmidhuber, J. Long short-term memory. *Neural computation 9*, 8 (1997), 1735–1780.

[60] Hubel, D. H., and Wiesel, T. N. Receptive fields of single neurones in the cat's striate cortex. *The Journal of physiology 148*, 3 (1959), 574–591.

[61] Hubel, D. H., and Wiesel, T. N. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology 160*, 1 (1962), 106–154.

[62] Hubel, D. H., and Wiesel, T. N. Receptive fields and functional architecture of monkey striate cortex. *The Journal of physiology 195*, 1 (1968), 215–243.

[63] Huber, J., Sheik-Nainar, M., and Matic, N. Force-enabled touch input on the steering wheel: An elicitation study. In *Proceedings of the 9th International Conference on Automotive User Interfaces and Interactive Vehicular Applications Adjunct* (2017), pp. 168–172.

[64] Hughes, D., Profita, H., and Correll, N. Switchback: An on-body rf-based gesture input device. In *Proceedings of the 2014 ACM International Symposium on Wearable Computers* (New York, NY, USA, 2014), ISWC '14, ACM, pp. 63–66.

[65] Hughes, D., Profita, H., Radzihovsky, S., and Correll, N. Intelligent rf-based gesture input devices implemented using e-textiles.

[66] IBM Cloud Education. Convolutional neural networks. [Online; accessed October 22, 2021].

[67] Jan, D., and Zeevaert, L. *Receptive multilingualism: Linguistic analyses, language policies and didactic concepts*, vol. 6. John Benjamins Publishing, 2007.

[68] Jarden Applied Materials. *RESISTAT®TYPE F901, MERGE D044*, 12 2014.

[69] Jones, L., Nabil, S., McLeod, A., and Girouard, A. Wearable bits: scaffolding creativity with a prototyping toolkit for wearable e-textiles. In *Proceedings of the Fourteenth International Conference on Tangible, Embedded, and Embodied Interaction* (2020), pp. 165–177.

[70] Kahveci, T., and Singh, A. Variable length queries for time series data. In *Proceedings 17th International Conference on Data Engineering* (2001), IEEE, pp. 273–282.

[71] Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., and Fei-Fei, L. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition* (2014), pp. 1725–1732.

[72] Keil. Arm cmsis-dsp. `https://arm-software.github.io/CMSIS_5/DSP/html/modules.html`.

[73] Keogh, E., Chakrabarti, K., Pazzani, M., and Mehrotra, S. Locally adaptive dimensionality reduction for indexing large time series databases. In *Proceedings of the 2001 ACM SIGMOD international conference on Management of data* (2001), pp. 151–162.

[74] Koenderink, J. J. The structure of images. *Biological cybernetics 50*, 5 (1984), 363–370.

[75] Korzeniewska, E., and Krawczyk, A. Applications of smart textiles in electromedicine. In *2019 19th International Symposium on Electromagnetic Fields in Mechatronics, Electrical and Electronic Engineering (ISEF)* (2019), IEEE, pp. 1–2.

[76] Krueger, R. A., and Casey, M. A. Designing and conducting focus group interviews, 2002.

[77] Laptev, I., Caputo, B., Schüldt, C., and Lindeberg, T. Local velocity-adapted motion events for spatio-temporal recognition. *Computer vision and image understanding 108*, 3 (2007), 207–229.

[78] LeCun, Y., Bengio, Y., et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks 3361*, 10 (1995), 1995.

[79] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE 86*, 11 (1998), 2278–2324.

[80] LeCun, Y., et al. Generalization and network design strategies. *Connectionism in perspective 19* (1989), 143–155.

[81] Lee, K. J., Roldan, W., Zhu, T. Q., Kaur Saluja, H., Na, S., Chin, B., Zeng, Y., Lee, J. H., and Yip, J. The show must go on: A conceptual model of conducting synchronous participatory design with children online. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (2021), pp. 1–16.

[82] Levenshtein, V. I. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady* (1966), vol. 10, pp. 707–710.

[83] Lin, J., Keogh, E., Lonardi, S., and Chiu, B. A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery* (2003), pp. 2–11.

[84] Lindeberg, T. Scale-space for discrete signals. *IEEE transactions on pattern analysis and machine intelligence 12*, 3 (1990), 234–254.

[85] Lindeberg, T. Scale-space theory: A basic tool for analyzing structures at different scales. *Journal of applied statistics 21*, 1-2 (1994), 225–270.

[86] Lindeberg, T. Scale-space.

[87] Lindeberg, T. *Scale-space theory in computer vision*, vol. 256. Springer Science & Business Media, 2013.

[88] Lindeberg, T. Scale selection.

[89] Lowe, D. G. Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE international conference on computer vision* (1999), vol. 2, Ieee, pp. 1150–1157.

[90] Lowe, D. G. Distinctive image features from scale-invariant keypoints. *International journal of computer vision 60*, 2 (2004), 91–110.

[91] Lowe, D. G. Method and apparatus for identifying scale invariant features in an image and use of same for locating an object in an image, Mar. 23 2004. US Patent 6,711,293.

[92] Mann, S. Humanistic computing:" wearcomp" as a new framework and application for intelligent signal processing. *Proceedings of the IEEE 86*, 11 (1998), 2123–2151.

[93] McDonald, D. Q., Vallett, R., Solovey, E., Dion, G., and Shokoufandeh, A. Knitted sensors: Designs and novel approaches for real-time, real-world sensing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 4*, 4 (2020), 1–25.

[94] Mlakar, S., and Haller, M. Design investigation of embroidered interactive elements on non-wearable textile interfaces. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (2020), pp. 1–10.

[95] Myers, C., Rabiner, L., and Rosenberg, A. Performance tradeoffs in dynamic time warping algorithms for isolated word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing 28*, 6 (1980), 623–635.

[96] Nabil, S., Jones, L., and Girouard, A. Soft speakers: Digital embroidering of diy customizable fabric actuators. In *Proceedings of the Fifteenth International Conference on Tangible, Embedded, and Embodied Interaction* (2021), pp. 1–12.

[97] Nanjappan, V., Shi, R., Liang, H.-N., Lau, K. K.-T., Yue, Y., and Atkinson, K. Towards a taxonomy for in-vehicle interactions using wearable smart textiles: insights from a user-elicitation study. *Multimodal Technologies and Interaction 3*, 2 (2019), 33.

[98] Niennattrakul, V., and Ratanamahatana, C. A. On clustering multimedia time series data using k-means and dynamic time warping. In *2007 International Conference on Multimedia and Ubiquitous Engineering (MUE'07)* (2007), IEEE, pp. 733–738.

[99] Olwal, A., Starner, T., and Mainini, G. E-textile microinteractions: Augmenting twist with flick, slide and grasp gestures for soft electronics. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (2020), pp. 1–13.

[100] Ortega, D. H., Cibrian, F. L., and Tentori, M. Bendablesound: a fabric-based interactive surface to promote free play in children with autism. In *Proceedings of the 17th International ACM SIGACCESS Conference on Computers & Accessibility* (2015), pp. 315–316.

[101] Parzer, P., Perteneder, F., Probst, K., Rendl, C., Leong, J., Schuetz, S., Vogl, A., Schwoediauer, R., Kaltenbrunner, M., Bauer, S., and Haller, M. Resi: A highly flexible, pressure-sensitive, imperceptible textile interface based on resistive yarns. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2018), UIST '18, ACM, pp. 745–756.

[102] Parzer, P., Probst, K., Babic, T., Rendl, C., Vogl, A., Olwal, A., and Haller, M. Flextiles: a flexible, stretchable, formable, pressure-sensitive, tactile input sensor. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (2016), pp. 3754–3757.

[103] Parzer, P., Sharma, A., Vogl, A., Steimle, J., Olwal, A., and Haller, M. Smartsleeve: Real-time sensing of surface and deformation gestures on flexible, interactive textiles, using a hybrid gesture detection pipeline. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2017), UIST '17, ACM, pp. 565–577.

[104] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in pytorch.

[105] Patton, M. Q. *Qualitative evaluation and research methods*. SAGE Publications, inc, 1990.

[106] Pointner, A., Preindl, T., Mlakar, S., Aigner, R., and Haller, M. Knitted resi: A highly flexible, force-sensitive knitted textile based on resistive yarns. In *ACM SIGGRAPH 2020 Emerging Technologies* (New York, NY, USA, 2020), SIGGRAPH '20, Association for Computing Machinery.

[107] Post, E. R., Orth, M., Russo, P. R., and Gershenfeld, N. E-broidery: Design and fabrication of textile-based computing. *IBM Syst. J. 39*, 3-4 (July 2000), 840–860.

[108] Poupyrev, I., Gong, N.-W., Fukuhara, S., Karagozler, M. E., Schwesig, C., and Robinson, K. E. Project jacquard: Interactive digital textiles at scale. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2016), CHI '16, ACM, pp. 4216–4227.

[109] Räihä, K.-J. Some applications of string algorithms in human-computer interaction. In *Algorithms and applications*. Springer, 2010, pp. 196–209.

[110] Reas, C., and Fry, B. Processing: Programming for the media arts. *AI Soc. 20*, 4 (Aug. 2006), 526–538.

[111] Romeny, B. M. H. *Front-end vision and multi-scale image analysis: multi-scale computer vision theory and applications, written in mathematica*, vol. 27. Springer Science & Business Media, 2008.

[112] Rumelhart, D. E., Hinton, G. E., Williams, R. J., et al. Learning representations by back-propagating errors. *Cognitive modeling 5*, 3 (1988), 1.

[113] Sakoe, H., and Chiba, S. Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing 26*, 1 (1978), 43–49.

[114] Schneegass, S., and Voit, A. Gesturesleeve: Using touch sensitive fabrics for gestural input on the forearm for controlling smartwatches. In *Proceedings of the 2016 ACM International Symposium on Wearable Computers* (New York, NY, USA, 2016), ISWC '16, ACM, pp. 108–115.

[115] Schneegass, S., and Voit, A. Gesturesleeve: using touch sensitive fabrics for gestural input on the forearm for controlling smartwatches. In *Proceedings of the 2016 ACM International Symposium on Wearable Computers* (2016), pp. 108–115.

[116] Schulz, K. U., and Mihov, S. Fast string correction with levenshtein automata. *International Journal on Document Analysis and Recognition 5*, 1 (2002), 67–85.

[117] Scovanner, P., Ali, S., and Shah, M. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th ACM international conference on Multimedia* (2007), pp. 357–360.

[118] Se, S., Lowe, D., and Little, J. Vision-based mobile robot localization and mapping using scale-invariant features. In *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)* (2001), vol. 2, IEEE, pp. 2051–2058.

[119] Seguine, R., et al. Capacitive sensing techniques and considerations. *Cypress Semiconductor Corporation Mobile Handset DesignLine* (2007).

[120] Senin, P. Dynamic time warping algorithm review. *Information and Computer Science Department University of Hawaii at Manoa Honolulu, USA 855*, 1-23 (2008), 40.

[121] Siyam, N., and Abdallah, S. A pilot study investigating the use of mobile technology for coordinating educational plans in inclusive settings. *Journal of Special Education Technology* (2021), 01626434211033581.

[122] Soegaard, M., and Dam, R. F. The encyclopedia of human-computer interaction. *The encyclopedia of human-computer interaction* (2012).

[123] Spinuzzi, C. The methodology of participatory design. *Technical communication 52*, 2 (2005), 163–174.

[124] Sullivan, P. Multiple methods and the usability of interface prototypes: the complementarity of laboratory observation and focus groups. In *Proceedings of the 9th annual international conference on Systems documentation* (1991), pp. 106–112.

[125] Sundholm, M., Cheng, J., Zhou, B., Sethi, A., and Lukowicz, P. Smart-mat: Recognizing and counting gym exercises with low-cost resistive pressure sensing matrix. In *Proceedings of the 2014 ACM international joint conference on pervasive and ubiquitous computing* (2014), pp. 373–382.

[126] Sutskever, I., Vinyals, O., and Le, Q. V. Sequence to sequence learning with neural networks. *arXiv preprint arXiv:1409.3215* (2014).

[127] Takamatsu, S., Kobayashi, T., Shibayama, N., Miyake, K., and Itoh, T. Meter-scale surface capacitive type of touch sensors fabricated by weaving conductive-polymer-coated fibers. In *2011 Symposium on Design, Test, Integration Packaging of MEMS/MOEMS (DTIP)* (May 2011), pp. 142–147.

[128] Tappert, C. C., Suen, C. Y., and Wakahara, T. The state of the art in online handwriting recognition. *IEEE Transactions on pattern analysis and machine intelligence 12*, 8 (1990), 787–808.

[129] Toews, M., Wells III, W., Collins, D. L., and Arbel, T. Feature-based morphometry: Discovering group-related anatomical patterns. *NeuroImage 49*, 3 (2010), 2318–2327.

[130] Trekhleb, Oleksii. Levenshtein distance. [Online; accessed October 11, 2021].

[131] Tuli, A., Chopra, S., Singh, P., and Kumar, N. Menstrual (im) mobilities and safe spaces. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (2020), pp. 1–15.

[132] Vallett, R., McDonald, D. Q., Dion, G., Kim, Y., and Shokoufandeh, A. Toward accurate sensing with knitted fabric: Applications and technical considerations.

[133] Vallett, R., Young, R., Knittel, C., Kim, Y., and Dion, G. Development of a carbon fiber knitted capacitive touch sensor. *MRS Advances 1*, 38 (2016), 2641–2651.

[134] Vallett, R. J. *A Differential Capacitive Sensing System for Knitted Textiles*. Drexel University, 2020.

[135] Vasseur, J.-P., and Dunkels, A. Chapter 1 - what are smart objects? In *Interconnecting Smart Objects with IP*, J.-P. Vasseur and A. Dunkels, Eds. Morgan Kaufmann, Boston, 2010, pp. 3–20.

[136] Vatavu, R.-D., Anthony, L., and Wobbrock, J. O. Gestures as point clouds: a $ p recognizer for user interface prototypes. In *Proceedings of the 14th ACM international conference on Multimodal interaction* (2012), pp. 273–280.

[137] Vazquez, V., Cardenas, C., Cibrian, F. L., and Tentori, M. Designing a musical fabric-based surface to encourage children with autism to practice motor movements. In *Proceedings of the 6th mexican conference on human-computer interaction* (2016), pp. 1–4.

[138] Vlachos, M., Kollios, G., and Gunopulos, D. Discovering similar multidimensional trajectories. In *Proceedings 18th international conference on data engineering* (2002), IEEE, pp. 673–684.

[139] Waterman, M. S., Smith, T. F., and Beyer, W. A. Some biological sequence metrics. *Advances in Mathematics 20*, 3 (1976), 367–387.

[140] Weiser, M. The computer for the 21 st century. *Scientific american 265*, 3 (1991), 94–105.

[141] Wicaksono, I., and Paradiso, J. Knittedkeyboard: Digital knitting of electronic textile musical controllers.

[142] Wimmer, R., and Baudisch, P. Modular and deformable touch-sensitive surfaces based on time domain reflectometry. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2011), UIST '11, ACM, pp. 517–526.

[143] Wu, T., Fukuhara, S., Gillian, N., Sundara-Rajan, K., and Poupyrev, I. Zebrasense: A double-sided textile touch sensor for smart clothing. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (2020), pp. 662–674.

[144] Wu, T.-Y., Qi, S., Chen, J., Shang, M., Gong, J., Seyed, T., and Yang, X.-D. Fabriccio: Touchless gestural input on interactive fabrics. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (2020), pp. 1–14.

[145] Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., and Bengio, Y. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning* (2015), pp. 2048–2057.

[146] Yang, J., Nguyen, M. N., San, P. P., Li, X. L., and Krishnaswamy, S. Deep convolutional neural networks on multichannel time series for human activity recognition. In *Twenty-fourth international joint conference on artificial intelligence* (2015).

[147] Yi, B.-K., and Faloutsos, C. Fast time sequence indexing for arbitrary lp norms.

[148] Yin, A. L., Gheissari, P., Lin, I. W., Sobolev, M., Pollak, J. P., Cole, C., and Estrin, D. Role of technology in self-assessment and feedback among hospitalist physicians: Semistructured interviews and thematic analysis. *Journal of medical Internet research 22*, 11 (2020), e23299.

[149] Zeiler, M. D., and Fergus, R. Visualizing and understanding convolutional networks. In *European conference on computer vision* (2014), Springer, pp. 818–833.

[150] Zhang, H., Tao, X., Wang, S., and Yu, T. Electro-mechanical properties of knitted fabric made from conductive multi-filament yarn under unidirectional extension. *Textile Research Journal 75*, 8 (August 2005), 598–606.

[151] Zhang, Y., Laput, G., and Harrison, C. Electrick: Low-cost touch sensing using electric field tomography. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2017), CHI '17, ACM, pp. 1–14.

[152] Zhou, B., Singh, M. S., Doda, S., Yildirim, M., Cheng, J., and Lukowicz, P. The carpet knows: Identifying people in a smart environment from a single step. In *2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)* (2017), IEEE, pp. 527–532.

[153] Zhu, M., Memar, A. H., Gupta, A., Samad, M., Agarwal, P., Visell, Y., Keller, S. J., and Colonnese, N. Pneusleeve: In-fabric multimodal actuation and sensing in a soft, compact, and expressive haptic sleeve. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (2020), pp. 1–12.