Department of Mathematics and Computer Science
Interconnected Resource-aware Intelligent Systems Research Group

Towards domain agnostic Wi-Fi CSI gesture classification models through table-to-image preprocessing and reinforcement learning

Master's Thesis

Yvan Putra Satyawan
Supervisors:
Your First Committee Member
Your second Committee Member
Your Third Committee Member

Draft version

Eindhoven, February 2023

Abstract

THIS IS MY ABSTRACT

Abstract

Preface

This goes out to all my supporters throughout this thesis, my supervisor, my friends, my family. Love you all.

actually write a proper preface

Contents

List of Figures

List of Tables

Listings

Chapter 1

Introduction

- Developments in IoT applications
- Developments in ubiquitous computing
- Developments in ubiquitous sensing and embedded sensors
- Challenges in a user interface for ubiquitous computing
- Wi-Fi sensing as a potential solution

With this thesis, we aim to explore the use of CNN architectures and domain-shift mitigation methods to improve the state-of-the-art in Wi-Fi CSI-based gesture classification.

1.1 Context and Background

- IoT devices and ubiquitous computing has become quite common and are convenient
- Main issue is how to provide some sort of always-available interface
- Sci-Fi presents gesture-based interfaces, but this requires ubiquitous sensors
- Issue with learning based approaches is that it often suffers from an inability to adapt to domain shifts
- In ubiquitous sensing, minimal setup is required on the user's part, otherwise it won't become mainstream if it is tedious to set up
- Why is Wi-Fi a potential solution to a low-cost ubiquitous sensing device
- 802.11bf, sensing standardization using Wi-Fi shows this might become common in the future and is being taken seriously by multiple large industry stakeholders

Wi-Fi technology, when boiled down, is just a really complex radio and what is radar but a different form of very complex radio. It naturally, or not so naturally, follows then, can Wi-Fi be used for remote sensing analogously to radar technology? The answer to this question, according to [1] and [2], the answer is a resounding yes!

1.2 Motivation

- Potential IoT/smart home applications
- To provide a ubiquitous sensing system which is low-cost and already prevalent in many environments
- To investigate and advance domain-agnostic learning systems
- To further the state-of-the-art for gesture classification
- If 802.11bf comes to fruition, and we want to take advantage of it, we should start now

1.3 Problem Statement

- The potential for Wi-Fi to be a modality for ubiquitous sensing should not be underestimated.
- Its prevalence in almost every modern building and home shows how widespread the technology already is.
- Domain-agnostic models exist, but are not as good as specialiized models
- The issue mostly boils down to the lack of generalizability of these existing models and the lack of ability to deal with domain-shift in any learning-based application
- What solutions may exist to improve performance

We aim to improve the state-of-the-art results in domain-agnostic gesture classification. We first clean the raw CSI signals using techniques common in radar technology and from the literature, providing a cleaner signal for the model to work with. We then use table-to-image transformations to allow for our model to use images as its input, utilizing advances in learning-based image-processing algorithms. Finally, we utilize a reinforcement learning domain-recognition approach to provide our classification model with a latent representation of the domain, providing it with additional information to improve its prediction performance.

Chapter 2

Literature Review

This is the first real chapter, this depends on your thesis structure.

Chapter 3

Second Real Chapter

And the second real chapter.

Bibliography

[1] Fadel Adib and Dina Katabi. “See through Walls with WiFi!” In: *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*. SIGCOMM ’13. Hong Kong, China: Association for Computing Machinery, 2013, pp. 75–86. ISBN: 9781450320566. DOI: 10.1145/2486001.2486039. URL: https://doi.org/10.1145/2486001.2486039.

[2] Kevin Chetty, Graeme E Smith, and Karl Woodbridge. “Through-the-wall sensing of personnel using passive bistatic wifi radar at standoff distances”. In: *IEEE Transactions on Geoscience and Remote Sensing* 50.4 (2011), pp. 1218–1226.

[3] Chip Huyen. *Designing Machine Learning Systems*. USA: O’Reilly Media, 2022. ISBN: 978-1801819312.

Appendix A

Plan

Complete by	Task
30 February	Initial infrastructure setup. This involves building the train-validate-test loops, building the data ingest/transformation pipelines, and building the model-building pipelines. This is feasible since much of the code will be taken from the IRIS Seminar project.
15 March	Integration of all modules as well as integration of hyperparameter optimization and training-tracking code complete. Initial training/debugging of the network can begin.
30 March	Parallelization of reinforcement-learning and deep-learning components of the network are complete. This involves making it possible to run the reinforcement-learning and the deep-learning models on separate computers during training, if this improves performance.
15 April	Initial results should be done by now, bugs will be found and "beta-testing" of the framework is in full swing. All results at this point are taken with a massive grain of salt since something will cause the results to be wrong, speaking from experience.
30 April	Initial good results should be available. Deeper investigations through hyperparameter optimization and changes to the model architecture should now start or is already ongoing.
30 May	Final results from the experiments should be completed.
30 June	First draft of the paper is done.
7 July	Second draft of the paper is done.
14 July	Final draft of the paper is done.
21-28 July	Somewhere in these two weeks, the thesis defense should take place.

Table A.1: Project timeline by task completion date

Table A.1 provides a timeline of the project and planning on when each part of the project should be completed by.

Appendix B

Risk Assessment

Risks identified for each phase of the plan and mitigation options are investigated in this section. Risk mitigation strategies are built from the author's previous experience working in an AI research lab and as a data scientist and software developer in industry. Additionally, strategies are also developed from theory learned in his computer science bachelor's.

Initial infrastructure setup During the initial infrastructure setup, the entire pipeline will be developed. This includes a modular data-ingest/transformation pipeline and model-building pipeline. Risks include data availability/usability and bugs in the data pipeline code. Additionally, the model will be completely modular and not prebuilt, increasing the chances of a bug appearing during this building phase but increasing flexibility of the model being investigated.

Data availability/usability refers to the fact that while the data is a public and published dataset, it is nonetheless quite large and it would not be feasible to download the entire dataset and place it on TU/e's HPC server. This means that some way to compress the data must be done. It is also possible that the transformed version of the data can be compressed more efficiently and this is what will end up being the dataset we work with for the majority of the project.

The data pipeline is also modular, allowing for data augmentation to be added "on-the-fly" instead of being hard-coded. This increases flexibility, but introduces the risk of bugs in unexpected circumstances. Mitigation factors include having written similar pipelines multiple times in the past and reuse of old, known-good code from the IRIS Seminar project and 2AMM10 Deep Learning course. Additionally, a similar approach has been use in previous research that we have completed, and we have significant experience in similarly modular pipelines.

Similarly, the model is built only at run-time, allowing for more flexibility and the possibility to fine-tune the model architecture using hyperparameter optimization techniques. This increases the chance that a good model architecture is chosen and strengthens the reasoning behind the chosen model architecture through empirical performance. Mitigation factors are the same as for the data pipeline.

Module Integration The most complex part of this infrastructure is to ensure that all modules work well together and there are no bugs in the hand-off step between modules.

To mitigate these factors, we take some advice from Chip Huyen's book Designing Machine Learning Systems [3]. To ensure that errors are not made during this integration, data flow will be closely monitored and visualized at every step through a UI, such that it is easy to see if anything went wrong at any step. The application of this will essentially be most of the interim steps being given some sort of output so we can visualize their result and track how data is transformed throughout the entire process.

Parallelization This is potentially unnecessary and may take up time that could be better used elsewhere. The idea is essentially that it might make sense to have the reinforcement-learning model and the gesture-classification model run on separate computers and having them communicate through some network.

Mitigation for this being unnecessary is providing only a limited amount of time to do this and the understanding that if this seems too difficult/may take too long, then we will immediately shelf the idea.

General bugs As with any software-based project there will inevitably be bugs in the code. Software engineering principle which lead to fewer bugs, such as proper use of debugging tools (but not test suites as we don't believe they will be necessary for a project with a limited scope such as this), will be used throughout work on this thesis. Additionally, use of "magic numbers" will be limited and as many parameters as appropriate will be assigned from variables.