# From Compression of Wearable-based Data to Effortless Indoor Positioning

**Lucie Klus**

Tampere University

Universitat Jaume I

Supervisors: Prof. Jari Nurmi (Tampere University)
Prof. Elena Simona Lohan (Tampere University)
Dr. Carlos Granell Canut (Universitat Jaume I)

This thesis has been completed in a joint/double Doctoral Degree programme at Tampere University, Finland and Universitat Jaume I, Spain.

Tampere (Finland) March 2023



# From Compression of Wearable-based Data to Effortless Indoor Positioning

**Doctoral Thesis**

**Lucie Klus**

Supervisors: Prof. Jari Nurmi (Tampere University)

Prof. Elena Simona Lohan (Tampere University)

Dr. Carlos Granell Canut (Universitat Jaume I)

**This thesis has been completed in a joint/double Doctoral Degree programme at Tampere University, Finland and Universitat Jaume I, Spain.**

**Tampere (Finland)**
**March 2023**

**From Compression of Wearable-based Data to Effortless Indoor Positioning**

**Report submitted by Lucie Klus in order to be eligible for a joint/double doctoral degree awarded by the Tampere University and Universitat Jaume I**


**Doctoral programme in Dynamic Wearable Applications with Privacy Constraints**

**Tampere University**


**European Joint Doctorate Marie Sklodowska-Curie in A Network for Dynamic Wearable Applications with Privacy Constraints (A-WEAR)**

**Universitat Jaume I – Doctoral School**

Lucie Klus

Lucie Klus
Digitally signed by Lucie Klus
DN: cn=Lucie Klus, c=FI, o=Tampere University, ou=Electrical Engineering, email=lucie.klus@tuni.fi
Date: 2023.04.05 19:11:04 +03'00'

Prof. Jari Nurmi

Jari Nurmi
Digitally signed by Jari Nurmi
DN: cn=Jari Nurmi, c=FI, o=Tampere University, ou=ITC Faculty, email=jari.nurmi@tuni.fi
Date: 2023.04.05 09:00:27 +03'00'

Prof. Elena Simona Lohan

Simona Lohan
Digitally signed by Simona Lohan
DN: cn=Simona Lohan, c=FI, o=Tampere University, ou=ITC, email=elena-simona.lohan@tuni.fi
Date: 2023.04.04 17:20:24 +03'00'

Dr. Carlos Granell Canut

CARLOS| GRANELL| CANUT
Firmado digitalmente por CARLOS|GRANELL|CANUT
Fecha: 2023.04.05 08:44:08 +02'00'

**Tampere, March 2023**

This dissertation is funded by the European Union's Horizon 2020 Research and Innovation programme under the Marie Skłodowska Curie grant agreement No. 813278, A-WEAR.



LUCIE KLUS

# From Compression of Wearable-based Data to Effortless Indoor Positioning

ACADEMIC DISSERTATION
To be presented, with the permission of
the Faculty of Information Technology and Communication Sciences of Tampere University, and of the Doctoral School of Universitat Jaume I
for public discussion at Tampere University,
Korkeakoulunkatu 1, 33720 Tampere, Finland, Sähkötalo SA207 S4 auditorio
at 12 o'clock noon, April 27th 2023.

ACADEMIC DISSERTATION

Tampere University, Faculty of Information Technology and Communication Sciences
Finland

Universitat Jaume I, Doctoral School
Spain

| | | |
|---|---|---|
| *Responsible supervisor* | Professor Jari Nurmi<br>Tampere University<br>Finland | |
| *Supervisor(s)* | Professor Elena Simona Lohan<br>Tampere University<br>Finland | |
| | Dr. Carlos Granell Canut<br>Universitat Jaume I<br>Spain | |
| *Pre-examiner(s)* | Dr. Christos Laoudias<br>University of Cyprus<br>Cyprus | Prof. Fernando Javier Álvarez Franco<br>Universidad de Extremadura<br>Spain |
| *Opponent(s)* | Dr. Tobias Feigl<br>FAU Erlangen-Nürnberg<br>Germany | Dr. Christos Laoudias<br>University of Cyprus<br>Cyprus |

The originality of this thesis has been checked using the Turnitin Originality Check service.



ISBN 978-952-03-2831-3 (print)
ISBN 978-952-03-2832-0 (online)
http://urn.fi/URN:ISBN:978-952-03-2832-0

2023

*Mojí mamince, tatínkovi, babičce a dědečkovi.*

# PREFACE

The work presented in this thesis was carried out during the years 2019-2022 at the Department of Electrical Engineering at Tampere University (TAU), Tampere, Finland, and at Instituto de Nuevas Tecnologías de la Imagen, Universitat Jaume I (UJI), Castellón de la Plana, Spain. This thesis has been completed in a joint/double Doctoral Degree program between TAU and UJI as a part of the A-WEAR project for which I gratefully acknowledge the funding we have received from the European Union's Horizon 2020 Research and Innovation programme under the Marie Skłodowska Curie grant agreement No. 813278, A-WEAR. Without this support, this work would have never been possible.

Furthermore, I would like to express my deepest gratitude to everyone, who has helped me to get to this point, supported me, and made me grow, both professionally and personally.

First and foremost, I would like to express my sincere gratitude to my thesis supervisors. Thank you, Prof. Jari Nurmi, for letting me learn from the best, for your unlimited support, and encouragement throughout my ventures as a doctoral student. Thank you, Prof. Elena Simona Lohan, for the academic guidance, for teaching me how to grow as a researcher, as well as for all the care and encouragement you gave me throughout the years. Thank you, Dr. Carlos Granell Canut,

for everything you did for me, your leadership, and your support. I would also like to extend my gratitude to Dr. Joaquín Torres-Sospedra (Ximo) for acting as my unofficial supervisor, for always supporting me in realizing my ideas, giving me invaluable feedback on my work, and helping me to see the paths ahead. Your guidance is highly appreciated. Furthermore, I would like to acknowledge and thank Dr. Jukka Talvitie for his support, fruitful discussions, and for letting me know about the existence of the A-WEAR project. Without you, I would have never been here. I would like to extend my thanks to Prof. Mikko Valkama, who provided me with motivation and guidance, and to my friend Asst. Prof. Bo Tan for continuous support, encouragement, and inspiration.

I would like to extend my recognition to my thesis pre-examiners Prof. Fernando Javier Álvarez Franco and Dr. Christos Laoudias, as well as to my opponents Dr. Tobias Feigl, and Dr. Christos Laoudias, for investing time, energy, and effort into evaluating my thesis and offering me invaluable feedback. I am honored that scientists of such magnitude read my work.

I would like to thank my colleagues and co-authors, with whom I had the pleasure to collaborate and learn from over the years. I would like to extend my thanks to all the staff of A-WEAR, TUNI, and UJI, who enabled me to successfully progress through my doctoral studies. I thank all other ESRs for creating a friendly and productive environment across the universities and for always supporting each other. Especially, I would like to thank Darwin for the fruitful collaboration that grew into a friendship. I would also like to extend my gratitude to Alex who has helped me, guided me, and always spoken his mind. To Olga and Nadia, for mutual support and for the memorable time spent together, to my friend Tomáš, as well as to Salwa, Asad, Pável, Viktoriia, and many others.

Furthermore, I would like to acknowledge my colleagues and friends, with whom I have spent countless good moments here in Tampere. Especially, I would like to thank Carlos, Ruben, Selahadin, Olga and Sofus, Anna, Daria, Hans, Kaan, Antoine, Maja and Pavel, and Thilini and Chathura. Most of all, I wish to express my love to Laura and Alberto, the kindest people I know, who over my time in Finland became my closest friends, as well as to Ana and Micael, whose friendship and humor I will always cherish. I would also like to thank many of my friends in Czechia, especially Daniel and Pet'a, for showing me that for friendship long distance does

not matter.

Above all, I am forever grateful to my family, especially my mom Gabriela and dad Ivo, my grandma Milena and grandpa Vlastimil, Robert, Lenka, Marcelka, Milan, Honza, and Martin. I thank them for their unwavering support over the years, for their love, warmth, and sacrifices they have made for me over the years. I thank my parents for bringing me into the world and making me the person I am today and my grandparents for always being there for me. Your love was the force and the reason that allowed me to get where I am today. Everything I do, I do for you.

Finally, I would like to thank Agata, whose memory will not be forgotten, Tom, Nadia, Hania, Pavel, and Danka, for taking me into their family and for giving me the purpose of my life, my husband Roman. Romi, I am grateful for every day I have with you, for your patience with me, your unconditional love and support, and for your kind and generous heart. You complement me in every aspect of my being, and I cannot imagine my life without you. I love you.

Tampere, March 2023
*Lucie Klus*

# ABSTRACT

In recent years, wearable devices have become ever-present in modern society. They are typically defined as small, battery-restricted devices, worn on, in, or in very close proximity to a human body. Their performance is defined by their functionalities as much as by their comfortability and convenience. As such, they need to be compact yet powerful, thus making energy efficiency an extremely important and relevant aspect of the system. The market of wearable devices is nowadays dominated by smartwatches and fitness bands, which are capable of gathering numerous sensor-based data such as temperature, pressure, heart rate, or blood oxygen level, which have to be processed in real-time, stored, or wirelessly transferred while consuming as little energy as possible to ensure long battery life. Implementing compression schemes directly at the wearable device is one of the relevant methods to reduce the volume of data and to minimize the number of required operations while processing them, as raw measurements include plenty of redundancies that can be removed without damaging the useful information itself.

This thesis presents a number of contributions in the field of compression of wearable-based data, mainly in areas of lossy compression techniques designated for the time series sensor-based data and positioning. In the scope of this work, two novel time-series compression techniques are proposed, namely Direct Lightweight Temporal Compression (DLTC) and Altered Symbolic Aggregate Approximation

(ASAX), which are specifically designed to address relevant challenges of modern wearable systems. As many of the modern wearables also possess localization capabilities critical for navigation, tracking, and monitoring applications, reducing the computational and storage demands for indoor positioning applications is the second addressed challenge. Performing the positioning task quickly and efficiently on all connected devices, including wearables, becomes crucial in industrial applications, eHealth, or security. As the localization technique of choice in Global Navigation Satellite System (GNSS) signal-obscured scenarios, positioning via fingerprinting proves a reliable and efficient solution, while arising new challenges to be solved. Improving the efficiency of the fingerprinting-based system by applying lossy compressions onto the training radio map is realized by proposing, implementing, and evaluating various novel dimensionality-reduction techniques.

This thesis proposes Element-Wise cOmpression using $K$-means (EWO$K$), a bit-level compression based on element-wise $k$-means clustering, radio Map compression Employing Signal Statistics (MESS), a sample-wise compression that extracts signal statistics based on their locations, as well as evaluates feature-wise methods Principal Component Analysis (PCA) and Auto-Encoder (AE) that transform fingerprints into low-dimensional representation.

The evaluation in the thesis shows the effectiveness of each compression scheme on 26 different datasets and provides the results achieved by combining the individual schemes together, accomplishing multi-dimensional radio map compression that sustains high positioning accuracy of the dataset, despite manyfold size reduction.

The processing requirements of the positioning system are further addressed by proposing a cascade of models that reduces the required search space of the algorithm. By combining numerous Machine Learning (ML) architectures, it is capable of further reducing the positioning time (and thus, positioning effort), while improving the positioning performance. The thesis further includes the introduction of an indoor positioning dataset collected by the author, denoted TUJI 1, a novel performance metric to evaluate the latency caused by the lossy compression, and several crucial adjustments to the distance metric calculations, generalizing their applicability.

The thesis provides novel insights into the compression of sensor-based, time-series data and into reducing the computational effort of the fingerprinting positioning schemes while introducing a relevant number of novel and efficient solutions beyond the State-of-the-Art.

# RESUMEN EXTENDIDO

En los últimos años, los dispositivos vestibles (*wearables* en inglés) se han convertido en un elemento indispensable de la sociedad. Se caracterizan por ser dispositivos pequeños y con limitaciones en el tamaño de sus baterías. Su uso no se limita únicamente a la superficie del cuerpo, sino que también pueden ser implantados en su interior para, por ejemplo, aplicaciones médicas. A la hora de definir su funcionalidad, sus capacidades técnicas como su comodidad juegan un papel crucial. Por esta razón, es de vital importancia que sean compactos a la vez que computacionalmente potentes. Esto hace que la eficiencia energética sea uno de los aspectos más relevantes de este tipo de dispositivos. El mercado de los *wearables* está hoy en día dominado por los relojes inteligentes y las pulseras de actividad física, los cuales son capaces de recoger un sinfín de indicadores a través de sus numerosos sensores, como por ejemplo temperatura, presión, frecuencia cardíaca, o niveles de oxígeno en sangre. Esta información tiene que ser procesada en tiempo real, almacenada, o transmitida de forma inalámbrica, de tal forma que conlleve el menor consumo energético posible para garantizar la mayor duración de las baterías. La implementación de técnicas de compresión de datos en los dispositivos *wearable* es uno de los métodos más importantes para reducir el volumen de los datos y por tanto, minimizar el coste computacional asociado a su procesamiento. La redundancia en los datos en bruto hace posible su compresión sin que esto suponga una pérdida significativa de información.

Esta tesis doctoral presenta una serie de contribuciones en el campo de la compresión de datos obtenidos a partir de *wearables* y, en particular, en el campo de la compresión de datos con pérdidas para series temporales, que es característica del tipo de datos que recogen los sensores de dichos dispositivos. Este trabajo propone dos técnicas originales para la compresión de datos llamadas *Direct Lightweight Temporal compression* (DLTC) y *Altered Symbolic Aggregate Approximation* (ASAX), las cuales han sido específicamente diseñadas para hacer frente a los desafíos presentes en los dispositivos *wearable*. La técnica ASAX representa un método de compresión sin latencia y ligero, diseñado específicamente para ser implementada en aplicaciones con requisitos de latencia críticos que requieren una respuesta inmediata del sistema a cualquier cambio en el flujo de datos. El segundo método de compresión, DLTC, permite comprimir los datos de manera sencilla, a la vez que minimiza la degradación de la información como resultado de la compresión y su posterior reconstrucción. El método DLTC permite llevar a cabo esta tarea con un gasto computacional reducido, con mínimo retardo y permite hallar un compromiso entre el grado de compresión y la pérdida de información.

Como los *wearables* permiten geolocalización, la cual es de vital importancia para aplicaciones de navegación, seguimiento y monitorización, la capacidad de reducir el coste computacional y de almacenamiento de dichos datos constituye el segundo desafío que trata de solucionar esta tesis doctoral. El conocimiento de la ubicación de los dispositivos dentro de la red móvil es de vital importancia en las redes de última generación, y requiere de una precisión menor que un metro en cualquier momento, lo cual incluye el complejo escenario en interiores (*indoor* en inglés) . Llevar a cabo la tarea de posicionamiento de manera rápida y eficiente es de vital importancia en aplicaciones industriales, de salud o seguridad. A pesar de que el posicionamiento basado en señales de radiofrecuencia es una alternativa eficaz y eficiente cuando el posicionamiento por GNSS no es posible, todavía hay muchos desafíos que resolver. Por ejemplo, la necesidad de tener un conjunto de medidas etiquetadas en los *wearables* y el coste asociado a su procesamiento durante la tarea de posicionamiento, lo cual supone un esfuerzo computacional y de almacenamiento a la hora de diseñar dichos dispositivos.

La mejora de la eficiencia del posicionamiento basado en señales de radiofrecuencia mediante técnicas de compresión de datos con pérdidas, aplicadas a la información del mapa de radio, es llevada a cabo mediante diferentes técnicas originales de reducción

de la dimensionalidad del conjunto de datos. Las tres dimensiones del mapa de radio que pueden reducirse son el número de muestras, el número de características que cada muestra representa, y el tamaño en bits de cada elemento (medida) registrada en el mapa.

La técnica de compresión de datos a nivel de bit propuesta, llamada *Element-Wise cOmpression using K-means* (EWOK), reduce el mapa de radio encontrando los valores más representativos utilizando un método innovador de implementación del algoritmo *K-means*, el cual permite eliminar el efecto de aleatoriedad del sistema. Las técnicas de reducción de la dimensionalidad que disminuyen el número de características del mapa de radio están basadas en el análisis de componentes principales (PCA) por sus siglas en inglés, y en una red neuronal sin supervisión llamada *Auto-Encoder* (AE).

Un método original de compresión llamado *radio Map compression Employing Signal Statistics* (MESS), reduce el número de muestras dentro del conjunto de datos de entrenamiento, de tal forma que encuentra una representación común de las medidas de posicionamiento en una localización común haciendo uso de sus estadísticas.

La evaluación proporcionada en esta tesis doctoral muestra la efectividad de cada uno de los métodos de compresión, en un total de 26 conjuntos de datos de acceso abierto, y también proporciona los resultados derivados de considerar los métodos individuales de manera conjunta. Esto permite conseguir una gran precisión en la localización a pesar de lograr reducir su tamaño en varios órdenes de magnitud.

Los requisitos de procesamiento para los sistemas de posicionamiento son reducidos gracias a *un modelo en cascada* que permite reducir el espacio de búsqueda del algoritmo. Esto se logra gracias a combinar diferentes arquitecturas de aprendizaje automático (*Machine Learning* o ML en inglés), lo cual permite reducir el tiempo de posicionamiento y, por lo tanto, el coste computacional, a la vez que mejora la precisión de la localización. Adicionalmente, esta tesis doctoral proporciona la base de datos del conjunto de medidas de localización en interiores, creada por la autora y que recibe el nombre de TUJI 1, una métrica original para evaluar la latencia asociada a los algoritmos de compresión con pérdidas y múltiples ajustes esenciales para calcular la distancia métrica, lo cual generaliza su aplicación.

Esta tesis doctoral proporciona ideas originales en el campo de la compresión de datos obtenidos a partir de sensores, de series temporales, y en la reducción del coste computacional del posicionamiento basado en señales de radiofrecuencia, a la vez que

proporcional numerosas soluciones eficientes y originales que van más allá del actual estado del arte.

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF ABBREVIATIONS

| | |
|---|---|
| **2D** | 2-Dimensional |
| **3D** | 3-Dimensional |
| **3GPP** | 3rd Generation Partnership Project |
| **5G** | Fifth Generation Mobile Networks |
| **AdB** | AdaBoost |
| **AE** | Auto-Encoder |
| **ALTC** | Adaptive Lightweight Temporal Compression |
| **AP** | Access Point |
| **APCA** | Adaptive Piecewise Constant Approximation |
| **AR** | Augmented Reality |
| **ASAX** | Altered Symbolic Aggregate Approximation |
| **BLE** | Bluetooth Low Energy |
| **CDF** | Cumulative Distribution Function |
| **CR** | Compression Ratio |
| **DBSCAN** | Density-Based Spatial Clustering of Applications with Noise |

| | |
|---|---|
| **DCT** | Discrete Cosine Transform |
| **DL** | Deep Learning |
| **DLTC** | Direct Lightweight Temporal Compression |
| **DT** | Decision Tree |
| **ECG** | electrocardiogram |
| **EWO*K*** | Element-Wise cOmpression using *K*-means |
| **FLAC** | Free Lossless Audio Codec |
| **GIF** | Graphics Interchange Format |
| **GNSS** | Global Navigation Satellite System |
| **iDCT** | inverse Discrete Cosine Transform |
| **IEEE** | Institute of Electrical and Electronics Engineers |
| **IoT** | Internet of Things |
| **IPS** | Indoor Positioning System |
| **JPEG** | Joint Photographic Experts Group |
| ***k*-NN** | *k*-Nearest Neighbors |
| ***K*-means** | *K*-means Clustering |
| **K-RLE** | K-Run-Length Encoding |
| **LAN** | Local Area Network |
| **LSTM** | Long-Short Term Memory |
| **LTC** | Lightweight Temporal Compression |
| **MAE** | Mean Absolute Error |
| **MESS** | radio Map compression Employing Signal Statistics |
| **ML** | Machine Learning |
| **MP3** | MPEG-1 Audio Layer III |
| **MSE** | Mean Squared Error |
| **NN** | Neural Network |
| **PAA** | Piece-wise Aggregate Approximation |
| **PCA** | Principal Component Analysis |

| | |
|---|---|
| **PLA** | Piece-wise Linear Approximation |
| **PNG** | Portable Network Graphics |
| **PPG** | photoplethysmogram |
| **ReLU** | Rectified Linear Unit |
| **RF** | Radio Frequency |
| **RFor** | Random Forest |
| **RLE** | Run-Length Encoding |
| **RLTC** | Refined Lightweight Temporal Compression |
| **RMSE** | Root Mean Squared Error |
| **RSS** | Received Signal Strength |
| **SAX** | Symbolic Aggregate Approximation |
| **SotA** | State-of-the-Art |
| **SD** | Standard Deviation |
| **SVM** | Support Vector Machine |
| **SVD** | Singular Value Decomposition |
| **UE** | User Equipment |
| **UWB** | Ultra Wide-Band |
| **VR** | Virtual Reality |
| **Wi-Fi** | IEEE 802.11 Wireless LAN |

# LIST OF SYMBOLS

| | |
|---|---|
| $\lvert\cdot\rvert$ | Cardinality (e.g. $\lvert\mathbf{S}\rvert$ number of samples in $\mathbf{S}$) |
| $\overrightarrow{l_i}$ | Ray from point $Z_j$, intersecting point $L_i$, i.e. lower bound |
| $\overrightarrow{u_i}$ | Ray from point $Z_j$, intersecting point $U_i$, i.e. upper bound |
| $\partial$ | Parameter defining the error tolerance |
| $\mathbf{R}$ | Set of reconstructed points after decompression, $\mathbf{R} = \{R_1, ..., R_{\lvert\mathbf{R}\rvert}\}$ |
| $\mathbf{S_{test}}$ | Set of testing samples |
| $\mathbf{S_{train}}$ | Set of training samples |
| $\mathbf{S}$ | Set of samples, $\mathbf{S} = \{S_1, ..., S_{\lvert\mathbf{S}\rvert}\}$ |
| $\mathbf{Z}$ | Set of origin points, $\mathbf{Z} = \{Z_1, ..., Z_{\lvert\mathbf{Z}\rvert}\}$ |
| $\mathcal{B}$ | Building |
| $\mathcal{F}$ | Floor |
| $f_{s,a}$ | The measured RSS value of $s^{th}$ sample at $a^{th}$ AP |
| $l_{i,d}$ | $d^{th}$ element of $i^{th}$ sample's label |

$\mathcal{N}$ Number of preserved coefficients

$\mathcal{T}$ Threshold

$\mu$ Mean

$\mu_{S_{train}/loc}$ Mean number of training samples per location

$\nu$ Delay [samples]

$\overline{(\cdot)}$ Reconstruction or estimate

$\overline{\mathcal{B}}_i$ Estimated building index of the $i^{th}$ sample

$\overline{f_{s,a}}$ The estimated RSS value of $s^{th}$ sample at $a^{th}$ AP

$\overline{\mathcal{F}}_i$ Estimated floor index of the $i^{th}$ sample

$\bar{l}_{i,d}$ $d^{th}$ element of $i^{th}$ sample's estimated label

$\psi_i$ Initialization method, where $i = \{max, min, xtr, imax, imin, ixtr\}$

$\sigma$ Standard Deviation

$\tau$ Time required for positioning of one sample [s]

$\tau_c$ Compression time [s]

$\tau_{train}$ Time required for training of model [s]

$\tilde{\tau}_\alpha$ Normalized time required for positioning of one sample to baseline $\alpha$ [-]

$\tilde{\tau}_\beta$ Normalized time required for positioning of one sample to baseline $\beta$ [-]

$\tilde{\varepsilon}_{2D\alpha}$ Normalized 2D positioning error to baseline $\alpha$ [-]

$\tilde{\varepsilon}_{2D\beta}$ Normalized 2D positioning error to baseline $\beta$ [-]

$\tilde{\varepsilon}_{3D\alpha}$ Normalized 3D positioning error to baseline $\alpha$ [-]

$\tilde{\varepsilon}_{3D\beta}$ Normalized 3D positioning error to baseline $\beta$ [-]

$\tilde{\zeta}_{\mathcal{B}\alpha}$ Normalized building-hit to baseline $\alpha$ [-]

$\tilde{\zeta}_{\mathcal{B}\beta}$ Normalized building-hit to baseline $\beta$ [-]

$\tilde{\zeta}_{\mathcal{F}\alpha}$ Normalized floor-hit to baseline $\alpha$ [-]

$\tilde{\zeta}_{\mathcal{F}\beta}$ Normalized floor-hit to baseline $\beta$ [-]

$\varepsilon_{2D}$ 2D positioning error [m]

$\varepsilon_{3D}$ 3D positioning error [m]

$\vee$ Logical OR operation

$\wedge$ Logical AND operation

$\zeta_{\mathcal{B}}$ Building-hit [%]

$\zeta_{\mathcal{F}}$ Floor-hit [%]

$ceil(\cdot)$ Rounding-up function

$CR$ Compression ratio [-]

$CR_{EWOk}$ Compression ratio of EWOk [-]

$CR_{pca}$ Compression ratio of PCA [-]

$CR_{tot}$ Compression ratio of combined compression methods [-]

$e_{MESS}$ Tunable parameter of MESS scheme

$K$ Number of clusters in $K$-means

$k$ Number of neighbors in $k$-Nearest Neighbors ($k$-NN)

$L_i$ Point on $\vdash_i$, $L_i = [x_{S,i}, y_{S,i} - \delta]$

$max(\cdot)$ Maximum value in set (e.g. $max(\mathbf{S})$ maximum value in $\mathbf{S}$)

$min(\cdot)$ Minimum value in set (e.g. $min(\mathbf{S})$ minimum value in $\mathbf{S}$)

$R_i$ $i$-th reconstructed samples, $R_i \in \mathbf{R}$

$RMSE$ Root Mean Squared Error expressed in the unit of data

$RMSE_{\%}$ Root Mean Squared Error expressed in [%]

$S_i$ $i$-th samples, $S_i \in \mathbf{S}$

$U_i$ Point on $\overset{\vdash}{u_i}$, $U_i = [x_{S,i}, y_{S,i} + \delta]$

$x_{S,i}$ $x$-axis element of sample $S_i$, $S_i = [x_{S,i}, y_{S,i}]$

$x_{Z,j}$ $x$-axis element of origin point $Z_j$, $Z_j = [x_{Z,j}, y_{Z,j}]$

$y_{S,i}$ $y$-axis element of sample $S_i$, $S_i = [x_{S,i}, y_{S,i}]$

$y_{Z,j}$ $y$-axis element of origin point $Z_j$, $Z_j = [x_{Z,j}, y_{Z,j}]$

$Z_j$ $j$-th origin point, $Z_j \in \mathbf{Z}$

CHAPTER 1

# INTRODUCTION

## 1.1 Thesis Motivation and Scope

In 2015, United Nations members agreed on 17 goals to transform our world into a better and more sustainable one, ensuring a better future for all [1]. These Sustainable Development Goals (SDGs) aim to end poverty, protect the planet, and ensure prosperity. They address critical topics such as "Zero hunger", good health and well-being, quality education, gender equality, as well as affordable and clean energy and sustainable cities and communities. Energy sustainability and efficiency of energy use are two of the most important factors to reach these goals. Similarly, 3rd Generation Partnership Project (3GPP) lists the decrease of power consumption of the Internet of Things (IoT) and wearables as the primary requirement for enabling the next-generation connections and networks [2].

This thesis aims to contribute to these goals and to provide means by which lengthy computations could be omitted, the number of executed operations minimized, and the amounts of stored data lowered. The thesis specifically focuses on energy savings within wearables, but the methods proposed here may also be implemented on other devices, ranging from small IoT devices to bigger and energy-

demanding machines.

Wearable devices are defined as small, battery-restricted devices, worn on or inside the (human) body. They range from small low-power gadgets such as those integrated into smart clothing or headphones, through ever-present smartwatches and fitness bands, to powerful and power-hungry Virtual Reality (VR) headsets. They can be classified based on the degree of attachment to the body into *accessories* including wrist-worn devices, smart rings, eTextiles, glasses, etc., *patches* that may be used for non-invasive medical monitoring, insulin regulation, continuous ECG measurement, glucose level assessment, etc., and *implants* ranging from small monitoring devices through more complex devices such as peacemakers, implantable defibrillators and even artificial organs.

The relevance of wearable devices in areas such as patient monitoring is nowadays acknowledged by both medical experts, and the patients themselves [3]–[5], including the older population [6]. Other areas of interest include industrial tracking [7], [8] and asset control, safety gadgets such as wearable carbon monoxide or radiation sensors [9], and entertainment devices such as Pokemon GO "Gotcha" wrist-worn catchers.

The wearable market is expected to keep on growing exponentially in the upcoming years and the amount of currently connected devices is estimated to be close to a billion. The growth of the market is to reach more than 150 billion EUR per year by 2028 [10].

There are numerous challenges connected to wearables, including a lack of unified standardization, addressing privacy and security, optimizing the performance and connectivity aspects, and boosting energy efficiency. The innovations presented within this thesis mainly address the latter, while focusing on preserving computational resources by reducing the amount of the data that have to be processed, stored, or transferred in the scope of sensor-based, time-series measurements and GNSS-free positioning systems. By addressing energy efficiency, the battery lifespan of the device gets extended, computational resources can be reallocated to different tasks, and consequently, novel functionalities can be enabled.

## 1.2 Thesis Objectives and Research Questions

The content of this thesis focuses on implementing novel approaches to reduce the computational load within wearable devices by implementing lossy compression approaches and model augmentations in relevant areas, with an emphasis on sensor-based data processing and indoor localization. By doing so, it aims to provide the means for accomplishing improved energy efficiency of wearable-based systems, with latency-free responses and lower computational demands as a basis for sustainable and ecology-friendly solutions.

To achieve these goals, this thesis aims to answer the following research questions:

RQ1. *Which lossy compression mechanisms can be implemented for energy-efficient, delay-sensitive wearable data gathering, transfer, and storage?*

RQ2. *To what extent can the bit-level, feature-wise, and sample-wise reduction of the radio map support accurate positioning while saving resources in data storage and transfer?*

RQ3. *How to compensate for k-NN's drawback of computationally expensive prediction on voluminous datasets?*

RQ4. *How to implement a multi-dimensional compression of the radio map to boost the performance efficiency of the positioning system?*

## 1.3 Author's Contributions and Thesis Structure

This thesis is a result of work conducted over a span of three years, starting in September 2019, under the umbrella of the A-WEAR project. The A-WEAR project is a training network under the European Union's Horizon 2020 (H2020) Marie Sklodowska-Curie Innovative Training Networks H2020-MSCA-ITN-2018 call, Grant Agreement no 813278. The thesis has been conducted as a part of a double/joint doctoral degree between Tampere University (TAU), Finland, and University Jaume I. (UJI), Spain entitled *Doctoral Programme in Dynamic Wearable Applications with Privacy Constraints*.

The work covered in this thesis is based on the Author's work in the fields of wearable technology [10], [11], time-series compression [12], [13], indoor positioning [14]–[17], outlier detection and data cleansing [15], [18], mobility management

optimization [19], [20], and model optimization and results' unification [17], [21]. Multiple extensions to the published work are also presented here for the first time.

Furthermore, a crucial part of the presented work has been based on the work that is currently under submission or will be submitted in a near future, including *EWOk: Towards Efficient Multidimensional Compression of Indoor Positioning Datasets*, which has been submitted to *IEEE Transactions on Mobile Computing* on 22nd November 2021 (currently under review). The Author of this thesis is the first author of this work.

The main contributions of this thesis are:

- deriving the crucial requirements for implementing compression schemes within wearable devices onto time-series, sensor-based data; pinpointing the relevant lossy compression schemes, and proposing novel ones to further boost the performance in scope of the requirements; evaluating the considered methods on real-world data (Chapter 2, addressed in [12], [13])
- introducing the relevant background for fingerprinting-based positioning systems and its parameters; introducing a novel positioning database gathered using heterogeneous devices; proposing a bit-level compression scheme denoted EWO$K$ and removing the effect of random initialization from $K$-means Clustering ($K$-means); implementing PCA and AE as the relevant radio map compression schemes; proposing MESS, a statistic-based method for sample-wise compression; providing thorough evaluation on 26 different datasets (Chapter 3, addressed in [14])
- introducing a cascade model for reducing the positioning time and effort; combining the aforementioned radio map compression schemes to achieve multidimensional compression and providing their robust evaluation; providing the summary of the individual advancements to the current State-of-the-Art introduced in each chapter (Chapter 4, addressed in [17])

The thesis is divided into an introduction, three main chapters, and a summary, and is organized in the following manner, highlighting the Author's main contributions:

Chapter 2 introduces the challenges concerning time-series data gathering and processing and introduces the concept of implementing lossy compression techniques on such data to boost the energy efficiency of the data processing. The main contri-

butions of this chapter include proposing two novel lossy compression methods and their thorough evaluation supplemented by proposing a new metric for the system latency assessment.

Chapter 3 introduces the basic concepts of an indoor positioning technique called fingerprinting and pinpoints the relevance of radio map compression approaches. The three dimensions of radio map compression are introduced, discussed, and novel techniques are developed to address bit-level, feature-wise, and sample-wise dimensionality reduction. Along with evaluating the proposed methods on 25 publicly available positioning datasets, a site survey was performed to gather an additional dataset that consists of measurements from multiple heterogeneous devices.

Chapter 4 addresses the challenge of reducing the computational effort of fingerprinting positioning systems by first proposing a cascade of models that splits the training database into smaller segments for lightweight inference while boosting the positioning accuracy. The chapter further combines the approaches introduced in Chapter 3 to achieve efficient, multi-dimensional radio map compression, capable of sustaining the positioning performance while preserving the computational and storage resources.

Furthermore, each chapter includes a conclusion discussing final remarks and take-away points, as well as an overview of the main contributions to the advancement of the current State-of-the-Art.

Chapter 5 concludes the main contributions of this thesis and introduces the Author's future work.

CHAPTER 2

# LOSSY COMPRESSION TECHNIQUES FOR ENHANCING THE ENERGY EFFICIENCY OF WEARABLE-BASED TIME-SERIES DATA

The recent progress in the field of wearable devices opens numerous unanswered research challenges and opportunities, including a search for novel computing paradigms, improving data transfer and processing efficiency, addressing privacy, or ensuring data security [10]. The majority of nowadays's wearables are smart, battery-operated machines [22] such as smartwatches, fitness trackers, wireless headphones, or smart patches, and require innovative common approaches in terms of extending their functionalities or processing capabilities while preserving or extending the battery lifespan [11]. The core limitations of wearable devices are their physical size, shape, and weight, as carrying them on user bodies has to be comfortable and effortless.

Consequently, increasing the device lifespan by increasing the battery size is not possible, thus preserving the battery by increasing the energy efficiency of the processes within the device is necessary.

One of the tasks that wearable devices enable is to monitor numerous biological and ambient signals, including electrocardiogram (ECG), respiration, or blood sugar monitoring, estimating heart rate using the photoplethysmogram (PPG) sensor or monitoring body and ambient temperature. Sensors in wearable devices collect large amounts of raw data, and smart ways to make the data volume smaller are critical in order to extend battery life. For comparison, PPG signals in wearables are recorded at up to 1 kHz frequency [23], and ECG signals at 500 Hz [24].

The relevant computing paradigms in the scope of wearable technologies, such as Multi-Access Edge Computing or Fog Computing [25], enable the offloading of heavy computations to more powerful devices. Nevertheless, the data itself still needs to be transmitted, costing valuable resources. Reducing the data volumes by applying compression schemes before transmission is a viable option in the scope of wearables [26]. Similarly, compression techniques enable efficient data mining of wearable data with significantly reduced effort [27].

In the scope of boosting the energy efficiency of a wearable device, compression techniques are capable of lowering the requirements for data transfer, processing, and storage, becoming an invaluable part of the system. Nevertheless, applying compression carries several challenges that need to be addressed. The primary challenge is to find a compression mechanism with low computational complexity so that running the compression algorithm itself consumes as little resources as possible in terms of processing power and operating memory. Similarly, the compression mechanism, especially if considering a lossy compression scheme, needs to perform its task of reducing the volumes of data without degrading the data quality in the process.

Lossy compressions are capable of greatly reducing the redundancies within the data when compared to lossless ones. Moreover, many of them have denoising capabilities, filtering the uncertainties within the raw measurements. Nevertheless, the higher the compression, the larger the information loss caused by the compression scheme. Finding the best-suited methods to achieve optimum compression trade-off is vital. Moreover, the compression scheme creates latency in the data stream, slowing down the reactiveness of the system to sudden changes, which is undesired, especially in e-Health or public safety applications. The three crucial requirements

on the compression schemes in the scope of wearable applications are depicted in Figure 2.1, and finding the optimal trade-off between them for each application often requires laborious testing and evaluation.

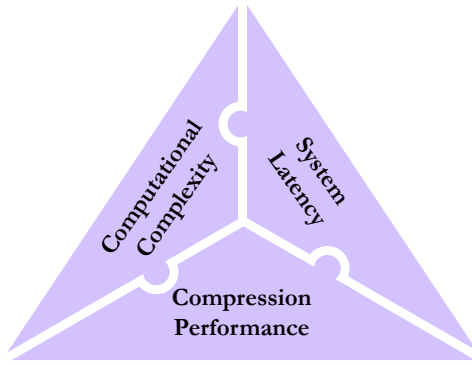


**Figure 2.1** Basic trade-offs and requirements on a compression scheme

This chapter focuses on finding novel solutions and providing recommendations for optimum lossy compression techniques across their applications onto wearable, sensor-based, time-series data by considering the criteria listed above represented by a number of performance metrics.

The chapter is structured as follows: Section 2.1 introduces the performance metrics and data used in the evaluation. Section 2.2 presents the relevant compression schemes across literature in the scope of wearable-based time-series sensor data processing and Section 2.3 introduces the novel compression schemes developed to further boost the compression performance, and overcome the shortcomings other the other State-of-the-Art (SotA) compression schemes. Section 2.4 provides the numerical results and their evaluation, Section 2.5 provides conclusions drawn from the chapter, and finally, Section 2.6 lists the Author's main contributions in the field.

## 2.1 Performance Metrics and Utilized Data

The techniques introduced in this chapter are developed for, and applied to, wearable-based, time-series sensor data. In order to demonstrate the characteristics and the capabilities of the considered lossy compression methods, both utilized in prior literature and the novel ones, four numerical performance metrics are considered.

The performance of each method is evaluated on three sets of data from publicly available datasets, which are introduced later in this section as well.

### 2.1.1 Performance Metrics

There are four main performance metrics utilized in this chapter. They serve as a base for comparison of the performance of the compression methods introduced in Sections 2.2 and 2.3.

#### Compression Ratio - CR

The primary function of lossy compression is to reduce the amount of data. The first metric, Compression Ratio (CR), represents the degree of compression in comparison with the original dataset. Across the literature, it is expressed in various forms, either in percentages or dimensionless. In this work, CR is defined as a ratio of the original (uncompressed) data to the compressed data size:

$$CR[-] = \frac{size(original\ data)}{size(compressed\ data)} \tag{2.1}$$

Thus, the CR higher than 1 signifies a reduced data size, e.g. CR= 10 means the data volume was reduced 10-fold, or by 90%.

#### Root Mean Squared Error - RMSE

The reconstruction error, characterizing the magnitude of difference between the original and reconstructed data after decompression, is commonly characterized using Mean Absolute Error (MAE), Mean Squared Error (MSE), or Root Mean Squared Error (RMSE) [28]. In this thesis, RMSE was chosen as the evaluation metric since it better captures the existence of large errors than MAE, yet is still expressed in the original units, unlike MSE, which squares it. The RMSE may be expressed as in Equation 2.2 in the same units as the data, or as in Equation 2.3 in percentage.

$$RMSE = \sqrt{\frac{1}{|\mathbf{S}|}\sum_{i=1}^{|\mathbf{S}|}[y_{S,i} - y_{R,i}]^2} \tag{2.2}$$

where $\mathbf{S} = \{S_1, ..., S_{|\mathbf{S}|}\}$ represents the input sample sequence of the length $|\mathbf{S}|$, $y_{S,i}$ represents the value on $y$-axis of $i$-th sample in this sequence and $y_{R,i}$ represents the $y$-axis value after the decompression of the corresponding reconstructed sample $R_i$, where $\mathbf{R} = \{R_1, ..., R_{|\mathbf{S}|}\}$.

$$RMSE_{\%} = \sqrt{\frac{1}{|\mathbf{S}|}\sum_{i=1}^{|\mathbf{S}|}[y_{S,i} - y_{R,i}]^2} \cdot \frac{100}{max(y_S) - min(y_S)} \tag{2.3}$$

where $max(\cdot)$ stands for the maximum value in the given set and $min(\cdot)$ stands for the minimum, $y_S$ is the set of $y$-axis values of samples $S$ in the set $\mathbf{S}$.

The advantage of considering Equation 2.3 lies in specifying the reconstruction error as a ratio, regardless of the unit. Such evaluation metric is then capable of comparatively evaluating the method when applied to two distinct sets of data.

### Compression Time - $\tau_c$

The compression time $\tau_c$ metric represents the effort required for compressing the data stream. It is expressed in seconds. It indirectly represents the complexity of the method by measuring the processing time that the algorithm requires in order to process the data and to output the compressed sequence. It can be assumed that the higher $\tau_c$ is, the higher the computational effort required from the system.

### Delay - $\nu$

The ability of each compression technique to react to a sudden change within the data is evaluated by considering the delay $\nu$. This delay expresses the maximum number of samples between the sample acquisition and the corresponding output's transmission, allowing its reconstruction. This metric is crucial when considering medical or safety control applications, as longer delay results in later data transmission, thus increasing the latency in the communication link. The delay is calculated specifically for each compression method, based on its given algorithm and therefore it is included before the numerical evaluation of considered methods, but after the introduction of the methods' algorithm, in Section 2.3.

### 2.1.2 Utilized Time-series Sensor Data

In order to evaluate the capabilities of the considered compression methods, two datasets were used. The first dataset called *An Open Dataset for Human Activity Analysis using Smart Devices* [29], [30] was used to obtain heart rate and atmospheric pressure data. This dataset was previously used in a number of research works across literature, including [31]–[33].

The original dataset consists of three datasets based on the device used for its acquisition, namely a watch, a phone, and glasses. For purposes of this work, only data measured by a watch are considered, as they are a representative of a wearable device. The watch brand was LG Urbane Watch 2 and the measured entities included a continuous heart rate measurement done by a PPG sensor, which is expressed in beats per minute - bpm (in total 91337 measurements), and atmospheric pressure measured by its barometer in kPa (14900 measurements). The distribution of the measurements, including the ranges they span over (the lines on $x$-axis represent the maximum and minimum values in respective datasets), may be found in their histograms (on the right), while the example of the data dynamics is demonstrated by their first 1000 samples on the left in Figures 2.2 and 2.3. By comparing the two sequences, atmospheric pressure data is significantly more noisy than the heart rate.

The data structure consists of a measured value and a timestamp for every measurement. The heart rate sensor frequency samples the data every approx. 0.2 s, while the barometer approx. every second. Despite the non-equidistant measurements in the raw data, we assume the periodic time frame for the evaluation, since several of the evaluated methods, especially Discrete Cosine Transform (DCT), include a buffer with a pre-defined size (DCT window). Transforming and reconstructing asynchronous data generates additional challenges, which could lead to a biased comparison of methods.

The second dataset utilized in this chapter is *ECG Heartbeat Categorization Dataset* [34] specifically developed for deep learning model training and evaluation. It combines two separate ECG datasets and pre-processes the original ECG sequences into short segments of the individual heartbeats, resulting in 109446 + 14552 = 123998 segments in total, each 188 samples long. The dataset is openly accessible on Kaggle repository [35].

For the evaluation, the first 50 segments from the testing set of the dataset were

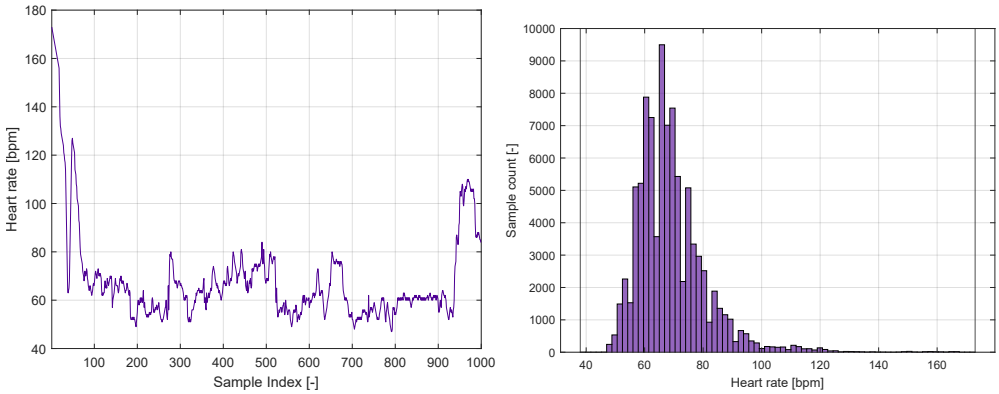


**Figure 2.2** Example of time-series and histogram of heart rate data

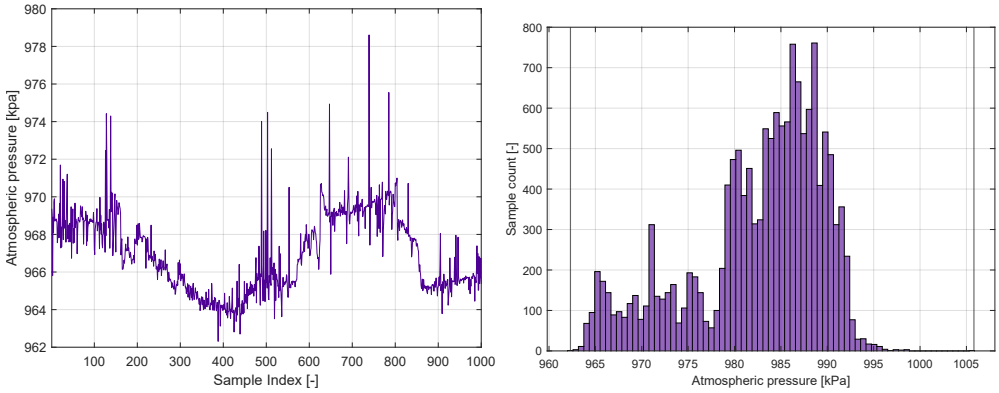


**Figure 2.3** Example of time-series and histogram of atmospheric pressure data

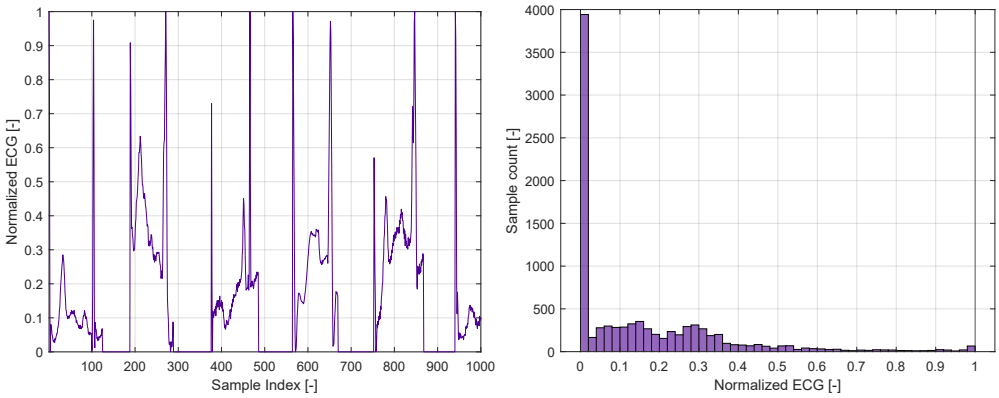


**Figure 2.4** Example of time-series and histogram of ECG data

used, each containing a heartbeat sample from across the dataset. In total, the combined sequence includes 9400 samples, which allows to sufficiently evaluate the individual methods in terms of adaptability to the dynamically changing data. The data dynamics are captured in Figure 2.4 - the first 1000 samples of the sequence are on the left, while the histogram of the whole series is on the right.

## 2.2 Relevant Compression Schemes for Data Transfer and Storage

The selection of the appropriate compression scheme strongly depends on the desired application, hardware and software capabilities of the device, as well as other parameters, such as the desired degree of compression, the maximum allowed information loss or the latency caused by the compression mechanism. This section presents the mechanisms and application of several popular lossy compression schemes from the literature, which are later used as benchmarks in the numerical evaluation.

### 2.2.1 Lossy Compression Essentials

The core objective of utilizing a compression mechanism is to reduce the size of data. The compression process can be lossless, when the full extent of the original data can be recovered, such as Graphics Interchange Format (GIF) and Portable Network Graphics (PNG) image formats or Free Lossless Audio Codec (FLAC) audio format. Alternatively there exist lossy compression schemes, which result in partial loss of original data during the compression, such as Joint Photographic Experts Group (JPEG) image format or MPEG-1 Audio Layer III (MP3) audio [36], [37].

On the one hand, the difference between the lossy and lossless compression, apart from the existence of the reconstruction error, is the achievable CR due to the tolerable information loss [28]. Lossless techniques are bound by the requirement of perfect data reconstruction, limiting their compression capabilities to removing statistical redundancies (e.g. Run-Length Encoding (RLE) [28], [38]) or incorporating information-preserving mechanisms, such as Huffman coding [38], [39], on a specific subset of data. On the other hand, lossy compression schemes alleviate the neces-

sity of preserving the full information in the process, creating countless innovative options of how to find the best way to maximize the CR, minimize the information loss caused by the compression, and ideally remove the undesired elements (e.g. noise) of the data in the process.

There are numerous lossy compression techniques suitable for time-series compression across current literature. The compression mechanisms range from transform-based methods, such as DCT, Fourier transform, or wavelet-transform compressions [38], through piece-wise aggregate methods such as Symbolic Aggregate Approximation (SAX) [40], K-Run-Length Encoding (K-RLE) [41] or Adaptive Piecewise Constant Approximation (APCA) [42], curve-fitting approximations such as Piece-wise Linear Approximation (PLA) or Lightweight Temporal Compression (LTC) [43], peaking and high-end, Deep Learning (DL)-based encoders [44], [45]. In the scope of wearable computing, many methods are unsuitable mostly due to their considerable computational complexity, memory requirements, or system latency created in the compression process. Among the available compression schemes, several methods were carefully selected with low complexity of the compression as a primary requirement. Furthermore, sample-wise processing methods are favored, as they process only a single sample at a time, preserving operating memory. Relevant studies comparing lossy compression techniques in the scope of sensor-based, wearable data include [44]–[48] or Author's prior evaluation [12], complementing the selection of the appropriate methods. ML-based methods, such as Neural Network (NN) encoders were not considered due to the inherent complexity, and additional practicalities such as the requirement of a dedicated decoder, data sensitivity, hyperparameter tuning, or the training process [45].

From the multitude of lossy compression techniques available across SotA literature, DCT was chosen as a representative of transform-based compression schemes utilized in our evaluation, SAX represents the piece-wise aggregate approximation category of methods, and curve-fitting methods are represented by LTC, which was developed as a lightweight and efficient piece-wise linear approximation alternative. Furthermore, we consider two LTC-derived methods as additional benchmarks.

### 2.2.2 Discrete Cosine Transform (DCT)

DCT is a function, which transforms the input array (or matrix) into spectral components, first introduced in [49]. Formally, it describes the input array in terms of the sum of cosines with varying frequencies. DCT and its properties were previously introduced and applied by the Author in [12]. It is widely used for multimedia data compression, such as image, sound or video (e.g. JPEG, MPEG). DCT has strong filtering properties, which in images may result in blurred edges, along with strong noise attenuation. In its essence, it is a simplified and generalized Fourier transform with purely real-valued outputs, as it performs the transformation based on decomposing the input array into multitudes of cosines. It offers strong de-noising properties.

This work considers DCT type 2 transformation from the algorithmic point of view, although other variants are available [50]. DCT-2 is the most commonly used variant across literature due to its performance.

DCT is not a compression scheme on its own, but rather a key operation in a compression scheme. After the transformation, many components include close-to-zero values. Zeroing insignificant components and clever aggregation of the zeros is the second key operation in DCT compression scheme. The zeroing mechanism is based on the maximum preserved energy of the original data sequence. After DCT transformation, the sum of squared frequency components represents the total energy within the data, and by iterative removal of the least significant (lowest-value) components and comparing the remaining signal energy with the original one, the desired preserved energy ratio can be obtained.

In this thesis, the aggregation is realized using a RLE technique. RLE iteratively aggregates the consecutive samples with the same $y$-axis values, returning a pair of numbers for each sequence, namely the given $y$-axis value and the sample count. In case the input sequence rapidly fluctuates, RLE can effectively result in increased sequence length than the input. In case RLE is combined with DCT, it effectively removes the excessive number of zero-valued spectral elements, which span across the majority of values after DCT compression.

Both RLE and DCT have straightforward inverse functions, which allow for efficient data reconstruction. Reversing the compression process of RLE compression is achieved by creating the reconstructed sample vector, within which each com-

pressed $y$-axis value is duplicated times defined by the corresponding entry in the sample count vector. The inverse operation of DCT, denoted inverse Discrete Cosine Transform (iDCT), transforms the individual spectral components back to the time-domain data sequence. Formally, the inverse transform to DCT-2 is denoted DCT-3 across the literature. The simplified diagram of this compression scheme combining DCT with RLE is depicted in Figure 2.5.

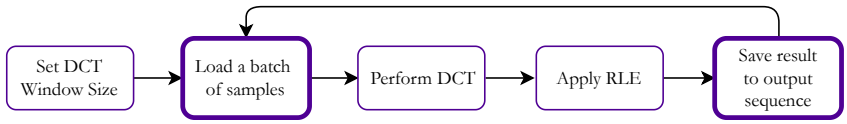


**Figure 2.5** Block diagram of DCT algorithm

DCT was utilized in the literature across numerous fields, spanning from radio map compression in [51], [52], image compression for classification [53], [54], or walk classification using wearable-based data [55].

One of the issues of implementing the DCT into a sensor-based system is its window size (buffer) parameter. In order to perform a DCT transformation the buffer needs to be filled with measurements first, thus the fluctuations in sampling intervals are lost. This complication is fully omitted in data streams such as video or speech, where the DCT window covers the frame and several consequent samples with constant data rate. Nevertheless, we include the DCT with RLE as one of the considered compression schemes due to the popularity of the method (further denoted as DCT).

### 2.2.3 Symbolic Aggregate Approximation (SAX)

Symbolic Aggregate Approximation (SAX) is a technique [40], [56] developed for reducing the dimensionality of the data in the time domain while indexing the symbols with lower-bounding distance measure. It is nowadays utilized for data mining purposes [57] as well while being extended to higher dimensional data [58]. The idea behind SAX is a two-step process, which is initialized with alphabetizing the vertical distance ($y$-axis) based on the original data distribution so that all alphabet symbols have approximately the same probability of occurrence (assuming Gaussian distribution). The first step of SAX data compression consists of applying a Piece-wise Aggregate Approximation (PAA) with a fixed frame onto the original data stream.

PAA simply aggregates the measurements within the frame (considered in a number of samples or in a time interval) and outputs a single value, namely the mean across the considered measurements. In the second step, each aggregated value is assigned to its closest alphabet symbol. Figure 2.6 depicts the whole process in a simplified algorithm.

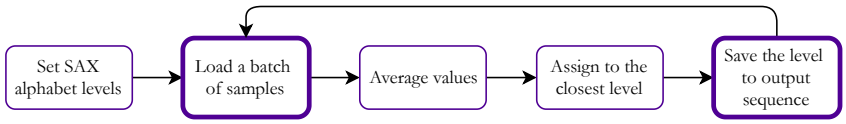


**Figure 2.6** Block diagram of SAX algorithm

The process is visualized in Figure 2.7, where the window size covers 3 samples (black dots). After their compression into one value (averaged pink lines), the alphabet symbol is immediately reported and sent. The alphabet size in the figure is 4 ($a$, $b$, $c$, and $d$ - purple dotted lines).

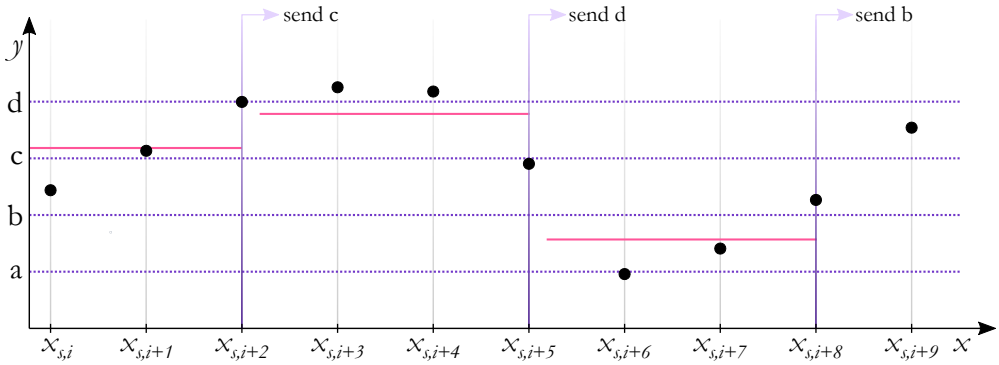


**Figure 2.7** Example of SAX on time-series

SAX as a method excels at reducing the dimensionality of the data and is robust to noise within the measurements due to PAA. It is also capable of synchronizing the data stream with non-equidistant sampling by decreasing the stream data rate. Nevertheless, the method is limited by its fixed-size window, which apart from denoising capabilities, also removes viable local relations within the useful part of data.

Across the literature, there are numerous SAX-derived methods including iSAX [57], adapting the method for large-scale data mining or extending SAX to multi-dimensional data in [58]. Trend Feature SAX [59] or Slopewise Aggregate Approximation [60] additionally capture the trend within the segment of data. A resource-

aware version of SAX additionally employs a clustering algorithm [61]. Furthermore, SAX, along with several SAX-derived methods were compared and evaluated in recent survey [62].

### 2.2.4 Lightweight Temporal Compression (LTC)

The Lightweight Temporal Compression (LTC) is a lossy compression method originally developed for the compression of microclimate data in [43], but soon gained attention across other fields, including sensor data [63], [64] or communications [65], [66], mainly for its straightforward implementation into sensor-based devices, such as wearable and IoT devices. Although [63] and [66] utilize LTC as a benchmark method, its performance is only outperformed in specific settings, proving its robustness across scenarios.

The algorithm of LTC method is depicted in Figure 2.8. LTC algorithm is initialized by setting the $\delta$-parameter, which defines the error tolerance of the method. The first sample in the input sequence is set as the first origin. After receiving the second sample, the two rays are created by intersecting the origin and two points, exactly $\delta$-distance above and below the second sample's $y$-coordinate, denoted the upper bound (above the sample) and the lower bound (below the sample). The subsequent sample is then obtained, and in case the $\delta$-range around the sample lies within or intersects the bounds, the new, reduced bounds are created. In case the $\delta$-range around the sample does not intersect the area between the bounds, the sample is considered out-of-bounds, and the algorithm initializes a cut-off sequence: first, the current origin point is saved to the output sequence, followed by creating new origin point mid-way between the upper and lower bounds, at the time instance of the last valid sample. Next, new bounds are constructed, originating form the new origin and intersecting the $\delta$-range above and below the out-of-bounds sample. The algorithm then repeats by loading the consecutive sample.

An example of the time-series sequence compression using LTC algorithm is captured in Figure 2.9, with marked instances of input data (black dots) with their $\delta$-ranges (pink line segments), out-of-bounds samples (red dots), origin points (green circles or dots - for newly made samples, that were not part of the original sequence), as well as upper bounds (dark purple lines), lower bounds (lilac lines), and data transmission instances.

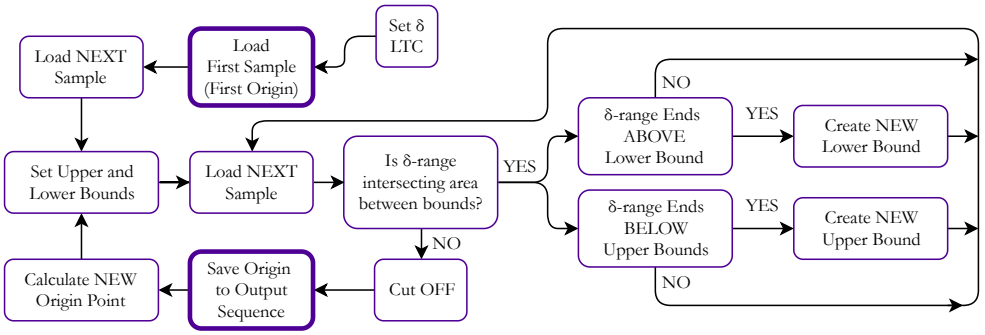


**Figure 2.8** Block diagram of LTC algorithm

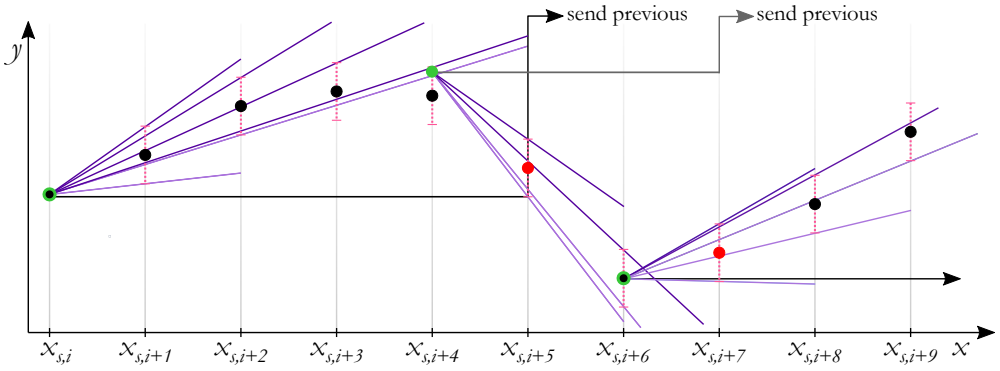


**Figure 2.9** Example of LTC on time-series

The main differences to DLTC are covered in Section 2.3. Further analysis and comparison of LTC performance to other methods is available in Section 2.4.

### 2.2.5 LTC-derived Methods

The introduction of LTC as a non-complex version of piece-wise linear approximation with innovative upper bound for the error led to proposing several LTC-derived methods in the following years. Among the proposed alterations, we consider Refined Lightweight Temporal Compression (RLTC) [67] and Adaptive Lightweight Temporal Compression (ALTC) [68] as the robust alternatives to the original method. The idea behind RLTC is creating a more robust and dynamic compression scheme, while the authors of ALTC aim for higher resilience against noisy measurements.

ALTC alters the algorithm of LTC to better represent the noise within the data by modifying the cut-off scheme. After finding a sample to be out-of-bounds, ALTC

creates the new origin point in the same way as LTC does at the $x$-coordinate of the last valid sample, immediately followed by considering the out-of-bounds sample as the next origin point, adding both to the output sequence. The new bounds are created only after receiving the next sample. The algorithm of ALTC achieves lower CR than the original method while sustaining the low complexity. The simplified algorithm is depicted in Figure 2.10. More detailed and additional information may be found in its original paper [68].

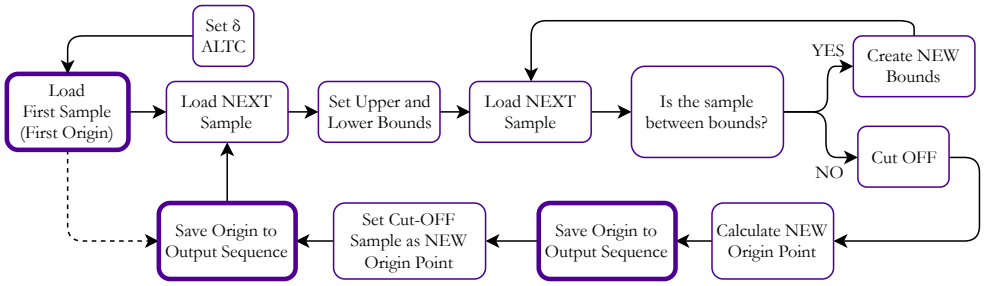


**Figure 2.10** Simplified block diagram of ALTC algorithm

The second considered LTC-derived method, denoted RLTC, aims to increase the overall robustness of the method by adding the tolerance bounds to the origin point as well. RLTC method creates additional origin points exactly $\delta$-range above and below the original one, then proceeds to create bounds for each bin separately. The cut-off is performed when all considered bins are out-of-bounds. The complexity of the method increases linearly with the number of created bins (as more $\delta$-ranges above and below the origin can be considered) while increasing the robustness of LTC to slowly-changing trends within the data stream. The simplified algorithm of RLTC is depicted in Figure 2.11, while more details regarding this method may be found in its original paper [67].

Additionally, LTC was extended to consider multi-dimensional data in [69] by considering intersections of convex cones. In 1D, this refers to the intersection of triangles, disks in 2D, and $n$-dimensional balls in the dimension $n$. The work offers derivation across arbitrary norms (e.g. norm 2 refers to Euclidean). Since LTC is developed to minimize the complexity, considering complex, multi-dimensional error bounds can be considered contra-productive. Instead, LTC (and all its derivatives) can consider each dimension separately, as 1D (plus time), and perform the cut-off across dimensions whenever one of the dimensions falls out-of-bounds.

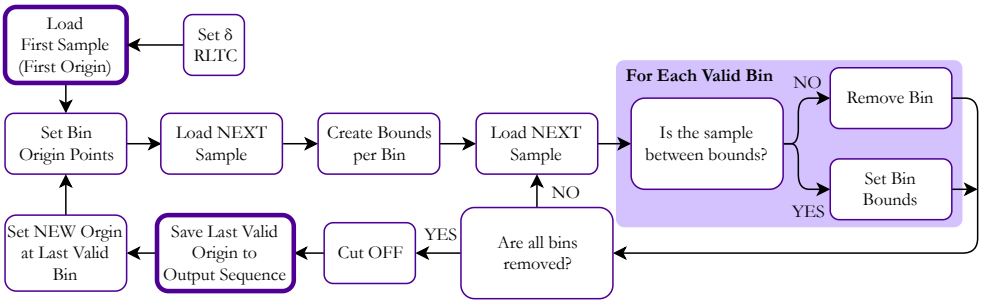


**Figure 2.11** Simplified block diagram of RLTC algorithm

## 2.3 Reducing Delays and Uncertainties in Wearable-based Systems

In Section 2.2, a number of compression schemes were introduced. They were chosen based on their wide utilization across SotA literature and their straightforward applicability to wearable devices. Nonetheless, there are numerous drawbacks to each of them, which can be compensated for in innovative ways. One such drawback is the system latency caused by their algorithms. In order to address the delay, as well as other shortcomings of the available methods, two compression schemes were proposed by the Author and are described in this section. Furthermore, the method-specific delay is addressed and described at the end of this section.

### 2.3.1 Proposed Altered Symbolic Aggregate Approximation (ASAX)

Altered Symbolic Aggregate Approximation (ASAX) is a simple, lossy compression method developed to minimize the compression delay, first introduced in [12]. ASAX is motivated by SAX and RLE methods, combining the discretization of the $x$-axis values and preserving the constant value at the output as long as it does not fluctuate.

The algorithm of ASAX is evaluated in a per-sample fashion, where the first input sample is assigned to its closest alphabet symbol level, followed by adding the alphabet symbol, along with its timestamp, to the output sequence. The next sample is loaded and assigned to the closest alphabet symbol level. If the sample's alphabet

symbol is equal to the previous sample's alphabet symbol, the sample is discarded. Otherwise, the new alphabet symbol, along with the last sample's timestamp are added to the output sequence. The simplified block diagram of ASAX is depicted in Figure 2.12.

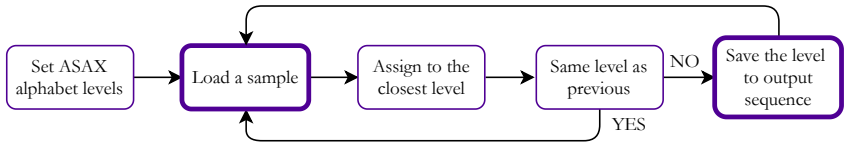


**Figure 2.12** Block diagram of ASAX algorithm

ASAX offers high flexibility in terms of setting the desired alphabet symbol levels, spanning from distribution-based solution, used in e.g. SAX, through offset-based, equidistant spacing, such as in K-RLE [41]. We propose the following alphabet level initialization method, creating equidistant alphabet bins across the data distribution.

1. Define the alphabet size $|\mathbf{A}|$
2. Get the *range* of $y$-axis values as $max(y) - min(y)$
3. Obtain the alphabet symbols levels as:

$$\mathbf{A} = min(y) + range \cdot \frac{(2 \cdot [1:1:|\mathbf{A}|]) - 1}{2 \cdot |\mathbf{A}|} \tag{2.4}$$

   Where $[1:1:|\mathbf{A}|]$ denotes a vector of values from 1 to $|\mathbf{A}|$ with an increment of 1.

The proposed alphabet initialization method ensures the smooth discretization of the values, regardless of the data distribution. Nonetheless, the alphabet levels might be adjusted based on a specific application.

Based on the studied literature, ASAX shares numerous similarities with APCA [42] and K-RLE [41]. APCA performs the piece-wise linear approximation of time-series sequence based on the $k$-NN search space, resulting in significantly higher complexity and compression delays, but flexible output sequence values and minimized RMSE. K-RLE compression's main difference to ASAX is the method's delay. K-RLE transmits the alphabet symbol with the number of instances it occurred in a row within the data, forcing the algorithm to wait for the change in alphabet symbol before transmitting the previous symbol. ASAX transmits the new symbol in

the same instance the change occurs. Additionally, ASAX offers better flexibility in selecting the alphabet symbols' levels.

The application of ASAX on a data stream is depicted in Figure 2.13. It shows the input samples (black dots) being assigned to the closest alphabet symbol's level (purple dotted lines) and immediately added to the output sequence. The alphabet size in the figure is 4 ($a$, $b$, $c$, and $d$), spanned equidistantly.

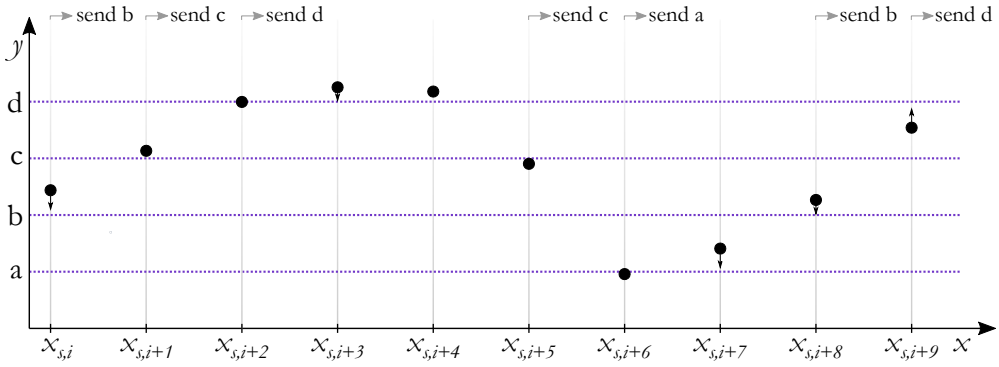


**Figure 2.13** Example of ASAX on a time-series

The method's primary purpose is to implement a latency-free data thinning algorithm, that dynamically captures any changes within the data. At any instance, the last value at the output of the algorithm can be interpreted as the current state of the monitored variable. In case the variable e.g. monitored heart rate changes, the algorithm immediately reacts by outputting a new alphabet symbol corresponding to the change. ASAX is specifically designed to be implemented in latency-free applications, that monitor constant or slowly-changing signals, capable of immediately reacting to sudden spikes and bumps within the data stream, such as heart rate, voltage, or flow monitoring, where a precise value may not be needed, but rather the change in a predefined zone is critical. As such, this method may be suitable for e.g. implementation into heart rate-tracking wearables, which store the results in the form of heart rate zones (1 to 5) [70].

### 2.3.2 Proposed Direct Lightweight Temporal Compression (DLTC)

Direct Lightweight Temporal Compression (DLTC) is a compression method developed in order to overcome some of the main drawbacks of the other previously described compression methods. It is a method based on LTC so that it keeps its main advantages - namely its low computational complexity and therefore good applicability to wearable and other sensor-based devices. The term "direct" represents the direct assignment of the input samples into the output sequence, as well as the immediate response of the system to a change in data. The method was first introduced by the Author in [13].

#### Direct Lightweight Temporal Compression - Algorithm

The algorithm of DLTC method is depicted in Figure 2.14. The key parameter of the methods, similarly to its predecessor LTC, is the so-called parameter $\delta$. It sets the maximum distance of the sample in sequence to its compressed value, and therefore, it is to be set as the maximum acceptable error in the given system. The higher the $\delta$-parameter is, the higher the error of reconstruction, but also the CR.

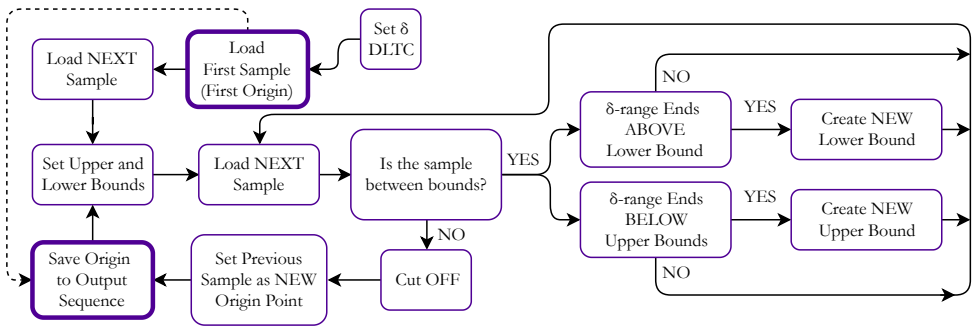


**Figure 2.14** Block diagram of DLTC algorithm

**The simplified description of the algorithm** is as follows:

When the $\delta$-parameter is set (the same for all samples in the given system), the first sample of the (to be compressed) sequence is loaded (the first origin point). It is immediately stored as the first sample of the output sequence and thus, it can be immediately transferred from the device, if needed. Next, the second sample of the

sequence is loaded, and upper and lower bounds are created so that the upper bound connects the origin point with the point in $\delta$-distance above the second sample's $y$-coordinate. Similarly, the lower bound is created by connecting the origin with the point in the $\delta$-distance below the second sample's $y$-coordinate. Next, another (third) sample in the sequence is loaded and compared with the upper and lower bounds. If this point lies in between bounds (it is considered a valid sample) the sample is used to create new upper and lower bounds (valid for the following sample). If this point is not in between the bounds, the cut-off process is initiated. In this case, the last valid sample is set as a new origin point and added to the output sequence (it may be immediately sent away) and the process of compression continues with setting the upper and lower bounds between the current original point and the points in $\delta$-distance above and below the cut-off point.

**The mathematical description of the DLTC algorithm** is as follows:

Let $\mathbf{S} = \{S_1, ..., S_{|\mathbf{S}|}\}$ be the time-series sequence, which is to be compressed. Let $S_i$, where $i = 1, ..., |\mathbf{S}|$ be the $i$-th input sample of this time-series, defined as $S_i = [x_{S,i}, y_{S,i}]$, where $x_{S,i}$ represents the index of the sample $S_i$ in time, and $y_{S,i}$ expresses the measured value. Additionally, let $\mathbf{Z} = \{Z_1, ..., Z_{|\mathbf{Z}|}\}$ be defined as the output sequence (a set of points to be transmitted and stored), where each point $Z_j$, where $j = 1, ..., |\mathbf{Z}|$ is the $j$-th output sample of the output sequence. Each of these points is defined as point $Z_j = [x_{Z,j}, y_{Z,j}]$, while $i \geq j$. The CR of DLTC ($CR_{DLTC}$) at any given instance $x_{S,i}$ is given as:

$$CR_{DLTC} = \frac{i}{j} \tag{2.5}$$

The size of CR is directly connected to the size of the tunable parameter $\delta$, which represents the maximum acceptable error in the compression process, as it sets the $\delta$-distance (value that is added/subtracted from the $y_{S,i}$) and consequently creates the bounds. Therefore, we define the size of $\delta$-range as double the value of $\delta$, spanning symmetrically around $S_i$ over the $y$-axis. The bigger the $\delta$-parameter is, the higher the final CR is. This parameter has to be defined before the beginning of the compression.

The pseudocode of DLTC algorithm is expressed in Algorithm 1 to support the understanding of the method.

The compression itself is initialized by loading the first time-series sample $S_1$ and setting it as the first output sample $Z_1$, so that $S_1 \triangleq Z_1$. Each sample $Z_j$ may be

**Algorithm 1:** Pseudocode for DLTC compression

**Input : S**, $\delta$
**Output: Z**

1 Initialization;
2 $j \leftarrow 1$ ;
3 $Z_1 \leftarrow [x_{S,1}, y_{S,1}]$ ;
4 $U_2 \leftarrow [x_{S,2}, y_{S,2} + \delta]$ ;
5 $L_2 \leftarrow [x_{S,2}, y_{S,2} - \delta]$ ;
6 $\overdash{u_2} \leftarrow get\_ray(Z_1, U_2)$ ;
7 $\overdash{l_2} \leftarrow get\_ray(Z_1, L_2)$ ;
8 **for** $i = 3 : length(\mathbf{S})$ **do**
9  **if** $[x_{S,i}, y_{S,i}]$ *is above* $\overdash{u_{i-1}}$ *or below* $\overdash{l_{i-1}}$ **then**
10  **Cut-off initialized:** ;
11  $j++$ ;
12  $Z_j \leftarrow [x_{S,i-1}, y_{S,i-1}]$;
13  $U_i \leftarrow [x_{S,i}, y_{S,i} + \delta]$ ;
14  $L_i \leftarrow [x_{S,i}, y_{S,i} - \delta]$ ;
15  $\overdash{u_i} \leftarrow get\_ray(Z_j, U_i)$ ;
16  $\overdash{l_i} \leftarrow get\_ray(Z_j, L_i)$ ;
17  **else**
18  **Reduce bounds:** ;
19  **if** $[x_{S,i}, y_{S,i}]$ *is below* $\overdash{u_{i-1}}$ **then**
20  $U_i \leftarrow [x_{S,i} y_{S,i} + \delta]$ ;
21  $\overdash{u_i} \leftarrow get\_ray(Z_j, U_i)$
22  **else**
23  $\overdash{u_i} \leftarrow \overdash{u_{i-1}}$
24  **end**
25  **if** $[x_{S,i}, y_{S,i}]$ *is above* $\overdash{l_{i-1}}$ **then**
26  $L_i \leftarrow [x_{S,i} y_{S,i} - \delta]$ ;
27  $\overdash{l_i} \leftarrow get\_ray(Z_j, L_i)$
28  **else**
29  $\overdash{l_i} \leftarrow \overdash{l_{i-1}}$
30  **end**
31  **end**
32 **end**

transferred from the compressing device immediately after being detected (lines 1 to 3 in Algorithm 1).

Next, the bounds are constructed. The upper bound $\overdash{u_i}$ is defined as a ray connecting the (current) output point $Z_j$, with the point $U_i$. This point is defined as a point in $\delta$-distance above the next point in the time-series (e.g. point $S_i$ or any following point in the time-series before the cut-off), therefore each point $U_i$ may be expressed as:

$$U_i = [x_{S,i},\ y_{S,i} + \delta] \tag{2.6}$$

Similarly, the lower bound $\overdash{l_i}$ is defined as a ray connecting the (current) output point $Z_j$, with the point $L_i$, which is defined as a point in $\delta$-distance below the next point in the time-series (e.g. point $S_i$). Similarly, each point $L_i$ is defined as:

$$L_i = [x_{S,i},\ y_{S,i} - \delta] \tag{2.7}$$

The definition of rays representing the lower $\overdash{l_i}$ and upper $\overdash{u_i}$ bounds, including points $U_i$ and $L_i$ that they intersect, is graphically expressed in Figure 2.15. The bounds are tied to point $S_i$, with previous sample $S_{i-1}$ representing the current origin sample, which also belongs to the output set (as point $Z_j$).

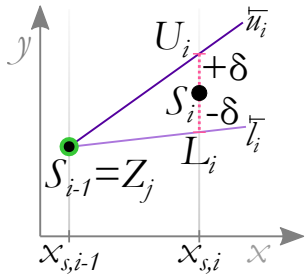


**Figure 2.15** Definition of DLTC bounds

When the first set of bounds is constructed (lines 4 to 7 in Algorithm 1), the algorithm processes each sample of a time-series in an iterative manner. First, the next point in the series is loaded (e.g. point $S_{i+1}$).

If the sample $S_{i+1}$ lies outside the area in between the bounds, so that the condition in Equation 2.8 is satisfied, the cut-off is initialized (line 9 in Algorithm 1).

$$y_{S,i+1} < \overdash{l_i}(x_{S,i+1}) \quad \vee \quad y_{S,i+1} > \overdash{u_i}(x_{S,i+1}) \tag{2.8}$$

Here, $\overdash{l_i}(x_{S,i+1})$ denotes the $y$-axis value of the ray $\overdash{l_i}$ at the time instance $x_{S,i+1}$, and

$\overset{\vdash}{u_i}(x_{S,i+1})$ denotes the $y$-axis value of the ray $\overset{\vdash}{u_i}$ at the time instance $x_{S,i+1}$. The symbol $\vee$ stands for the logical OR operation.

During the cut-off, the last valid sample $S_i$ is set as the new output point $Z_{j+1}$ and immediately added to the output sequence. The points $U_{i+1}$ and $L_{i+1}$ are constructed around the out-of-bounds point $S_{i+1}$ according to Equation 2.6 and Equation 2.7, followed by the creation of the new set of bounds $\overset{\vdash}{u_{i+1}}$ and $\overset{\vdash}{l_{i+1}}$, which are spanning from the point $Z_{j+1}$ through $U_{i+1}$ and $L_{i+1}$, respectively (lines 11 to 16 in Algorithm 1).

Otherwise, if the point $S_{i+1}$ lies within bounds and therefore fulfills the condition in Equation 2.9, a new set of bounds is created (line 17 in Algorithm 1).

$$\overset{\vdash}{l_i}(x_{S,i+1}) \quad \leq \quad y_{S,i+1} \quad \leq \quad \overset{\vdash}{u_i}(x_{S,i+1}) \tag{2.9}$$

In order to do so, the points $U_{i+1}$ and $L_{i+1}$ are constructed around the point $S_{i+1}$. If $U_{i+1}$ lies below $\overset{\vdash}{u_i}$, so that $U_{i+1} \leq \overset{\vdash}{u_i}(x_{S,i+1})$, the new upper bound $\overset{\vdash}{u_{i+1}}$ is constructed, spanning from the current output $Z_j$ through $U_{i+1}$ (line 19 to 21 in Algorithm 1). If $L_{i+1}$ lies above $\overset{\vdash}{l_i}$, so that $L_{i+1} \geq \overset{\vdash}{l_i}(x_{S,i+1})$, the new lower bound $\overset{\vdash}{l_{i+1}}$ is constructed, spanning from the current output $Z_j$ through $L_{i+1}$ (line 25 to 27 in Algorithm 1).

The algorithm then continues with loading the next sample of the input sequence (e.g. $S_{i+2}$) and repeating the process.

The example of a time-series compression using DLTC is depicted in Figure 2.16. The set of original data (black dots), each having displayed their $\delta$-ranges (pink vertical lines), is compressed using the mechanism explained above, where dark purple lines are used to denote the upper bounds and lilac lines denote the lower bounds. The red points represent the points initializing the cut-off (they are outside of the current (last) bounds), while the points in the green circle represent the output sequence points. These are transmitted immediately after the cut-off is detected.

**The reconstruction** of the data compressed using the DLTC method is realized using piece-wise linear interpolation. At any given time instance $x_i$, the reconstruction algorithm finds the closest preceding and succeeding samples in the output sequence $\mathbf{Z}$, further denoted as $Z_j$ and $Z_{j+1}$, where $x_{Z,j} < x_i \leq x_{Z,j+1}$. The reconstruction $y_i$ is obtained as the $y$-axis value of a line segment connecting $Z_j$ and $Z_{j+1}$ at the time instance $x_i$.

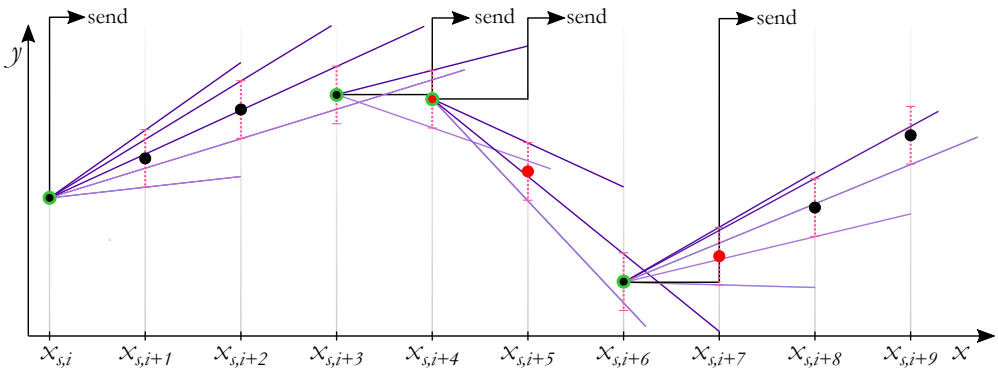


**Figure 2.16** Example of DLTC on a time-series

## The main advantages of DLTC over LTC

The novelty of DLTC, when compared to the traditional LTC, can be summarized as follows:

- The samples are added to the output sequence **Z** immediately after the cut-off is performed, resulting in a **reduced compression delay**. Each output sample's delay is reduced by the number of valid input samples between cut-offs, resulting in mean DLTC delay being lower exactly by the value of CR (in samples), assuming the same CR for LTC and DLTC.
- The condition for initializing the cut-off is satisfied when the sample itself is found outside the bounds, rather than the $\delta$-distance around the sample, as it is the case for LTC. Consequently, DLTC applies the error margin $\delta$ only once per iteration of the algorithm when creating the bounds, while LTC considers it in the cut-off decision as well. As the result, the CR of DLTC is lower than that of LTC at the same $\delta$ setting, while the **RMSE of DLTC is reduced and more stable.**
- DLTC assigns the last valid sample to the output sequence after the cut-off, while LTC considers the point between the upper and lower bounds at the last valid time instance, creating arbitrary data. On the one hand, this change slightly decreases the fitting capabilities of the original LTC, as the output sample's value is not affected by the intermediate samples, while on the other hand, adding the last valid sample to the output sequence enables smooth data trend monitoring, creating an undistorted origin point for the consecutive data

samples. After DLTC compression, the output sequence **Z** is composed of a subset of the input sequence **S** samples, making DLTC a **data-driven, computationally efficient redundancy reduction method.**

- All aforementioned novelties result in **reduced computational complexity** of the compression method. Adding samples directly to the output sequence preserves memory, while improved cut-off condition and direct assignment of the last valid sample reduce the amount of required algebraic operations within each loop.

DLTC is designed as a lightweight version of PLA, which does not interpolate between the samples, and rather applies a data-driven redundancy reduction without producing new samples. The method is suitable for a wide array of sensor-based data, characterized by piece-wise linear trends, such as ECG, glucose level monitoring, or temperature. The disadvantage of DLTC, as well as all LTC-derived methods, is their inability to distinguish noise from the desired signal, leading to the increased cut-off frequency on noisy data.

### 2.3.3 Method-specific delay

The inability of the compression methods to report the measured samples immediately after being measured is expressed as delay $\nu$, considered in the number of samples. This delay is caused by the algorithm, specific to each considered method. The delay is defined as the mean across the highest individual delays of the compression method, where the highest individual delay is the $x$-axis difference (in samples) between adding each compressed sample to the output sequence and the earliest input sample that allows its reconstruction.

Two of the studied methods (SAX and DCT) use a predefined fixed window size. Connected to the size of this window is its inevitable delay, which may be expressed (similarly for both DCT ($\nu_{DCT}$)) and SAX ($\nu_{SAX}$)) as:

$$\nu_{SAX/DCT} = window\ size - 1 \tag{2.10}$$

There is no delay caused by the ASAX method ($\nu_{ASAX}$). The method itself was created in order to avoid any lag in the system. At any given time instance, the last value reported by the compression mechanism system can be considered as current

and stays the same until a new level is reported, therefore:

$$\nu_{ASAX} = 0 \tag{2.11}$$

Based on the visualization in Figure 2.9, the j$^{th}$ set of input samples $S_i$ of LTC method can be defined as the input samples within an interval $\langle x_{z,j-1}, x_{z,j}\rangle$. The individual delay of j$^{th}$ set is then derived as:

$$\nu_{LTC/RLTC}(j) = (x_{z,j} - x_{z,j-1}) + (x_{z,j-1} - x_{z,j-2}) = x_{z,j} - x_{z,j-2} \tag{2.12}$$

The same Equation 2.12 is used to obtain the delay of j$^{th}$ set of input samples for the RLTC method ($\nu_{RLTC}(j)$). The delay $\nu_{xLTC}$ for all LTC-based methods is then obtained as the mean across the all sets of individual delays $\nu_{xLTC}(1)$ to $\nu_{xLTC}(|\mathbf{Z}|)$ :

$$\nu_{xLTC} = \frac{1}{|\mathbf{Z}|}\sum_{j=1}^{|\mathbf{Z}|}\nu_{xLTC}(j) \tag{2.13}$$

The delay of the proposed DLTC may be estimated similarly as the delay of LTC (see Figure 2.16) for each sample in j$^{th}$ set of samples as:

$$\nu_{DLTC/ALTC}(j) = x_{z,j} - x_{z,j-1} \tag{2.14}$$

The delay $\nu_{DLTC}$ is then obtained as a mean value over all sample-wise delays across the whole dataset as in Equation 2.13. The delay of the ALTC ($\nu_{ALTC}$) may be obtained the same as $\nu_{DLTC}$.

The definition of the delay consequently defines how the achieved compression affects the latency of the system. The delay $\nu$ is introduced, to our knowledge, for the first time as the performance metric in this thesis.

## 2.4 Numerical Analysis

In this section, the selected lossy compression techniques described in Sections 2.2 and 2.3 are evaluated in terms of the chosen performance metrics. The parameters of each method were swept to offer a valid range of possible trade-offs between compression efficiency, reconstruction error, delay, or compression time.

To support the theoretical pros and cons of each presented compression scheme,

a thorough evaluation process to assess each method's performance against the others is conducted and compared based on the theories introduced above. The evaluation focuses on the relevant trade-offs in terms of wearable technology, namely the reconstruction error RMSE against the achieved compression degree and the compression time against the CR. The evaluation of the achieved delay based on the CR is included. Moreover, the evaluation offers a numerical evaluation of each method based on the maximum reconstruction error, denoted RMSE threshold, which comparatively presents all performance metrics at once.

The evaluation is concluded in Section 2.5 by pinpointing the optimal lossy compression methods to optimize the data transmission and storage processes, boost system reactiveness to sudden changes, or reduce the algorithm complexity.

All evaluations were performed on a laptop computer with an Intel Core i7-8750H CPU and 32 GB of RAM. The software of choice was MATLAB 2021*b*.

### 2.4.1 Compression Performance Evaluation as CR and RMSE Trade-off

The general performance of the lossy compression method is best assessed based on its trade-off between achieved data size reduction and the information loss caused by the compression process. In order to thoroughly evaluate the given trade-off, we plot the achieved CR per each method against the reconstruction error RMSE achieved after the decompression. In the figures below, the more the function sequence leans toward the bottom right corner of the figure (high CR and low RMSE), the more favorable performance trade-off the compression method achieves.

Here, three DCT realizations were performed, with different DCT window sizes of 10, 25, and 100. The bigger the window size is, the bigger the complexity of the transformation and the higher the achievable compression ratio. The parameter swept in each realization of DCT was the maximum energy preserved in the sequence, as in practice, maximum preserved energy is a tunable parameter in the DCT compression scheme. Changing the DCT window re-designs the whole function in terms of required memory, processing power, and complexity. Adjusting it results in a different compression scheme. The capability of DCT compression is strongly defined by its window size. The highest achievable CR equals half of the window size when the output equals a single RLE pair.

The sweep of LTC-based methods varies based on the size of the $\delta$-parameter. SAX is swept across different window sizes and ASAX based on the alphabet size.

Figure 2.17 depicts the trade-off achieved on the heart rate data sequence. At CRs from 1 to 7, LTC, RLTC, and DLTC perform approximately at the same level, outperforming the other methods, yet when RMSE increases above 1%, DLTCs trade-off becomes more favorable. The results show that at CRs above 50, SAX begins outperforming DLTC in terms of the reconstruction error. Notably, the performance of LTC, RLTC, and ALTC declines at higher CRs. As the CR of DCT is limited by the window size, the maximum CR achieved by DCT with window size 100 approaches 50 (half the window size), with RMSE of 5.5%.

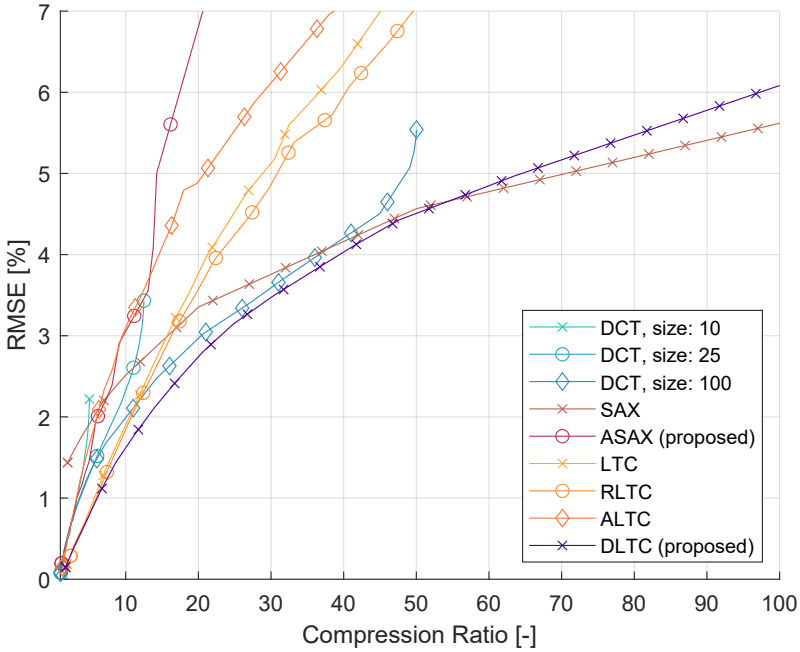


**Figure 2.17** Evaluation of CR and RMSE trade-off on heart rate data

The evaluation of atmospheric pressure data series is shown in Figure 2.18, where DCT outperforms the other methods at CRs above 20 and RMSE above 2.5%. Here, DCT sweep was performed at constant 0.99 preserved energy parameter over varying DCT window sizes to demonstrate the impact of increasing the DCT window. Note, that utilizing large-window DCT requires significantly higher computational performance. The sweep performed across different DCT window sizes enabled DCT to dynamically adapt to the data while decomposing a long sequence of sam-

ples at once. At lower CRs, DLTC outperforms the other methods. The results show, that lightweight compression techniques struggle when coping with uncertainties within the data stream. Interestingly, ALTC, as a variant of LTC developed for increased robustness on noisy data does not show much improvement over the original LTC.

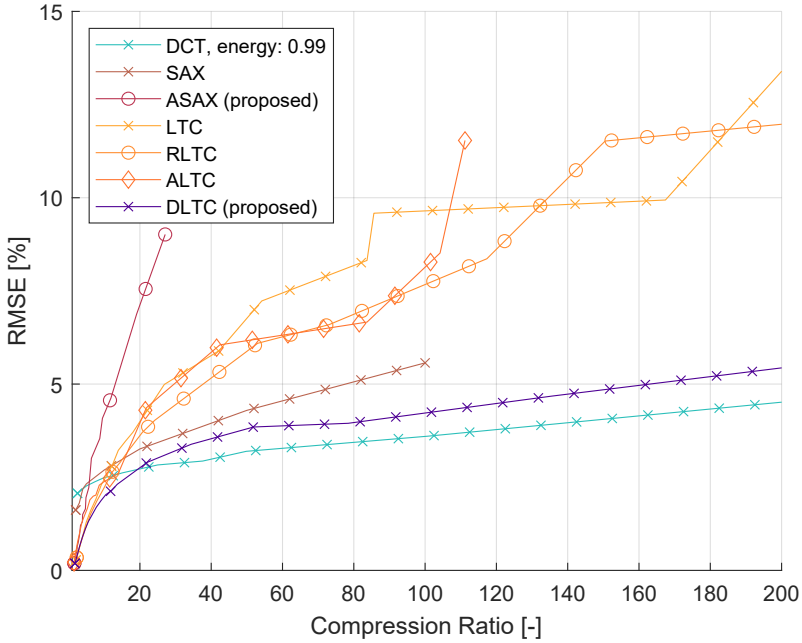


**Figure 2.18** Evaluation of CR and RMSE trade-off on atmospheric pressure data

The final set of CR and RMSE trade-off results on ECG data series shows DLTC method's dominance across all RMSE values in Figure 2.19. As a method developed for piece-wise linear data of which ECG is a perfect example, this evaluation shows that DLTC's changes to the LTC algorithm reduce RMSE, stabilize the trade-off sequence and allow for more accurate data reconstruction than all other considered LTC alternatives.

Overall, the CR and RMSE trade-off evaluation shows that the performance of each method depends on the selected method, its parameters, as well as the evaluation data itself. LTC-based methods show superior performance when considering highly dynamic data sequences with linear trends, such as ECG, while showing vulnerability in case the noise impairs the measurements. The evaluation on the atmospheric pressure sequence shows that DCT with considerable window size is capable

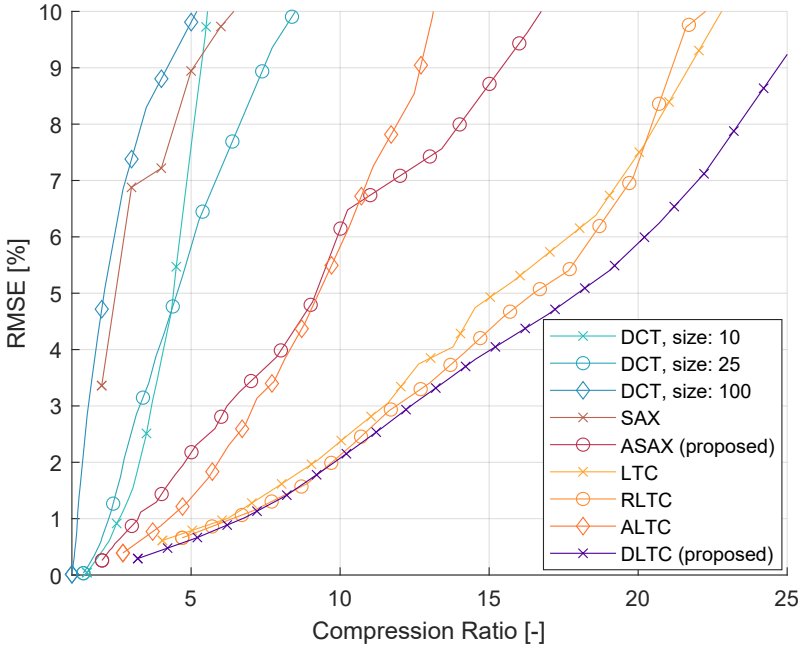


**Figure 2.19** Evaluation of CR and RMSE trade-off on ECG data

of preserving the uncertainties within the data at the expense of a large buffer and increased performance requirements, being able to efficiently cope with non-linearities. Across all sequences, DLTC provides the most consistent compression performance level among all considered lossy compression methods.

### 2.4.2 Compression Performance Evaluation as CR and $\tau_c$ Trade-off

This part of the numerical evaluation focuses on the trade-off between the achievable compression and the method's complexity in terms of compression time $\tau_c$. The sweep over individual methods' parameters remains unchanged.

Figure 2.20, Figure 2.21 and Figure 2.22 visualize the required compression time $\tau_c$ based on each method's achieved CR in the considered scenarios. The evaluation shows mostly consistent results across the figures, where SAX function achieves by far the lowest $\tau_c$ when processing the datasets, as the method performs the compression on full batches of data, as well as the compression itself is non-complex. The second fastest compression scheme is ASAX, which despite requiring sample-wise

processing, its algorithmic simplicity allows for weightless operation. Across the LTC-based methods, ALTC shows the most favorable performance, followed by LTC, and DLTC. RLTC operates significantly slower due to its higher-complexity algorithm. Figure 2.20 does not display the DCT with window sizes of 25 and 10, because their $\tau_c$ was much higher than that of the other methods (see the results in Table 2.1 further in this section).

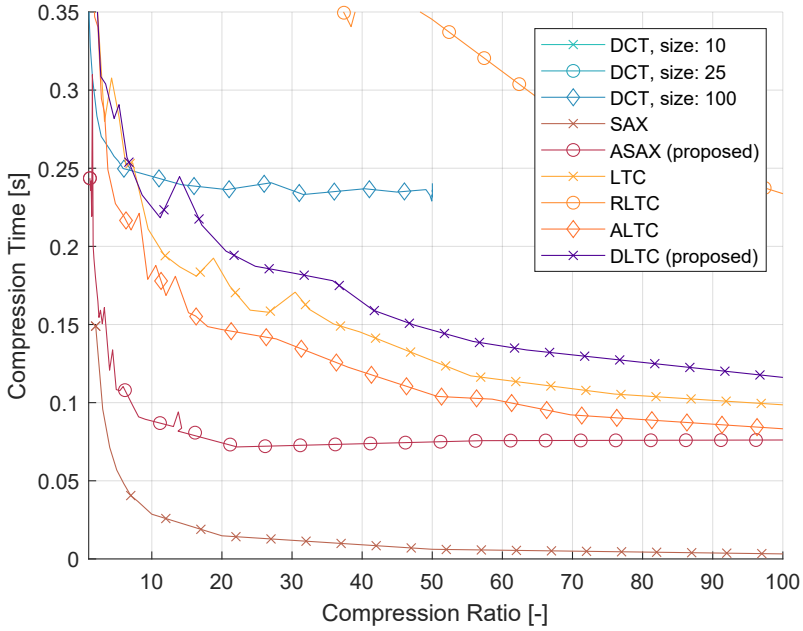


**Figure 2.20** Evaluation of CR and $\tau_c$ trade-off on heart rate data

The compression time $\tau_c$ of DCT strongly depends on the considered window size. As this evaluation was realized using MATLAB software on a high-performance machine, initializing the DCT function for each window of data resulted in higher computational effort than performing the DCT transformation itself, regardless of the window size, which would not be the case if performing the transformation on a low-power wearable device. Nevertheless, even large-window DCT sweep in Figure 2.21 was not able to clearly outperform the lightweight compression schemes.

Figure 2.22 does not display the CR-$\tau_c$ trade-off of the DCT with a window size of 10, as its compression time was significantly higher than that of the other methods (see the results in Table 2.3 further in this section). Based on the evaluation of CR and $\tau_c$ trade-off, SAX and ASAX create almost no computational strain on a device, followed by three LTC-based methods, namely LTC, ALTC, and DLTC.

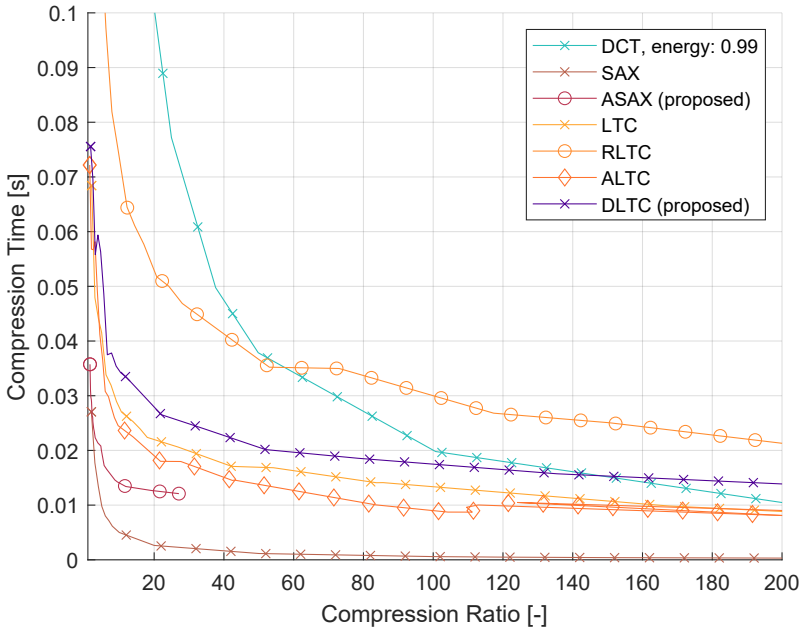


**Figure 2.21** Evaluation of CR and $\tau_c$ trade-off on atmospheric pressure data

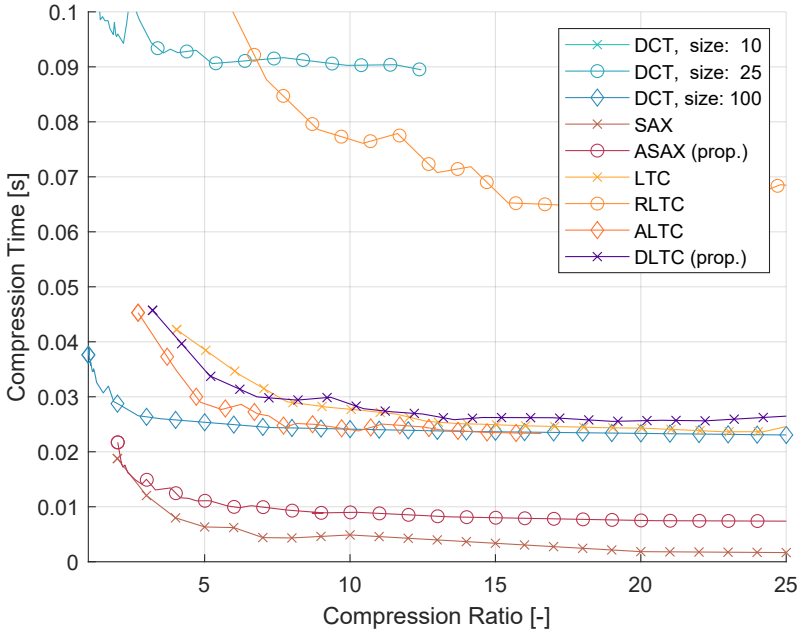


**Figure 2.22** Evaluation of CR and $\tau_c$ trade-off on ECG data

### 2.4.3 Compression Performance Evaluation as CR and Delay Trade-off

In the following paragraphs and figures, the novel performance metric, denoted delay $\nu$, is compared as a function of CR across the considered methods. For time-series compression techniques, the delay is a direct result of the performed compression and in the majority of cases increases linearly with the CR.

Figure 2.23 shows the behavior of delay on the heart rate data. The delay of ASAX method is equal to 0, regardless the CR, as all changes in the data stream are reported in the output data without latency (when considering delay $\nu$, the sample processing time is disregarded). The delay of DCT method is characterized by the window size, as its buffer defines the maximum delay of each batch of samples. The delay of a constant-window-size DCT is also constant, regardless of the achieved CR.

The delay of the remaining methods is linearly increasing with CR. DLTC, ALTC, and SAX are characterized by the delay-CR line with the slope $\sim 1$, or as $\nu \approx 1 \cdot CR$. The $\nu$ of SAX is defined by the window size (similarly to DCT), and the delays of DLTC and ALTC are defined by the offset-free cut-off algorithm (see Figure 2.16). The delay of LTC and RLTC methods are increasing the increasing CR with the slope of $\sim 2$, according to the numerical results. The increased slope of the delay is caused by the output sample offset in their respective algorithms (see Equation 2.12 for LTC and RLTC, which differentiates between $j^{th}$ and $(j-2)^{th}$ output samples).

Figure 2.24, visualizing the trade-off on atmospheric pressure data, confirms the previous findings. The only significant difference is caused by the DCT sweep over the window sizes. Now DCT's delay $\nu$ increases linearly with the CR with the slope of 2 (due to the overhead of lossless RLE).

The consecutive results on ECG data sequence comply with the previous findings and conclusions. Figure 2.25 also demonstrates that the delay-CR trade-off is independent of the data type, its dynamics, and uncertainties.

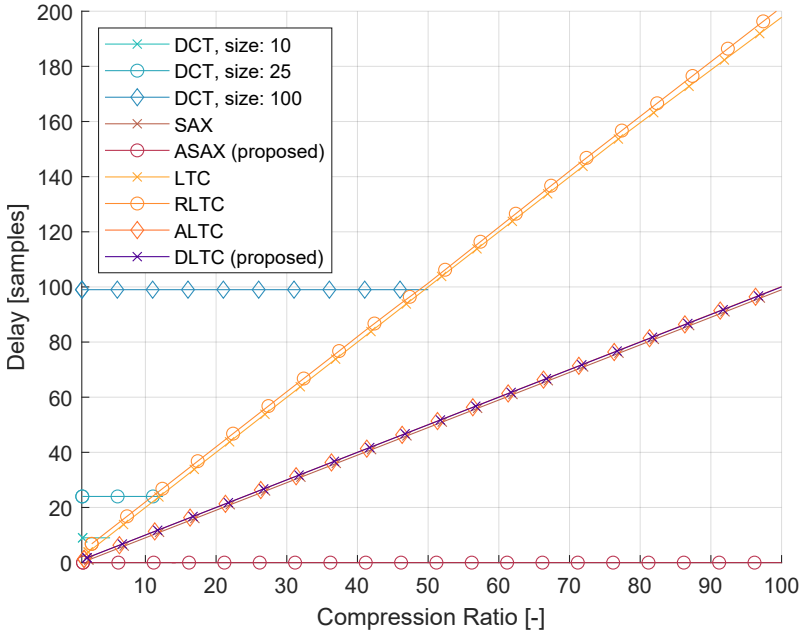


**Figure 2.23** Evaluation of CR and delay trade-off on heart rate data

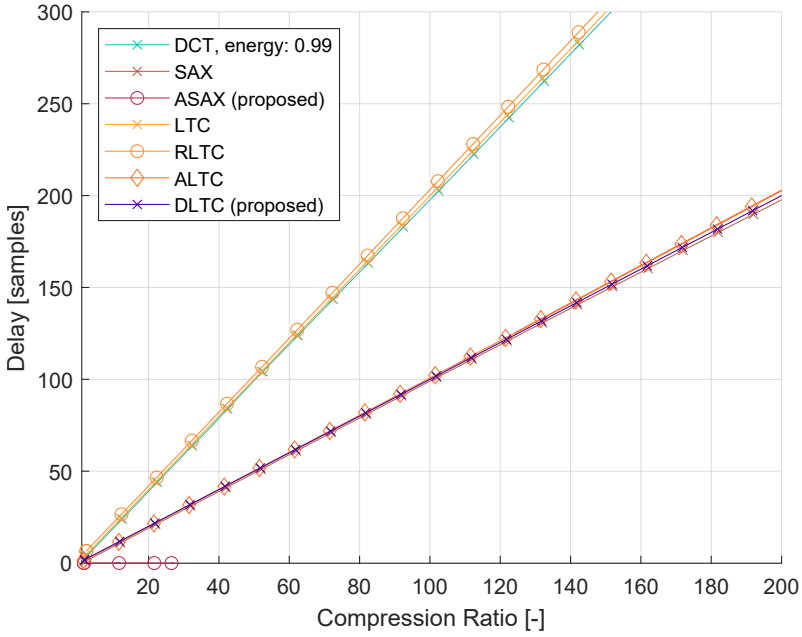


**Figure 2.24** Evaluation of CR and delay trade-off on atmospheric pressure data

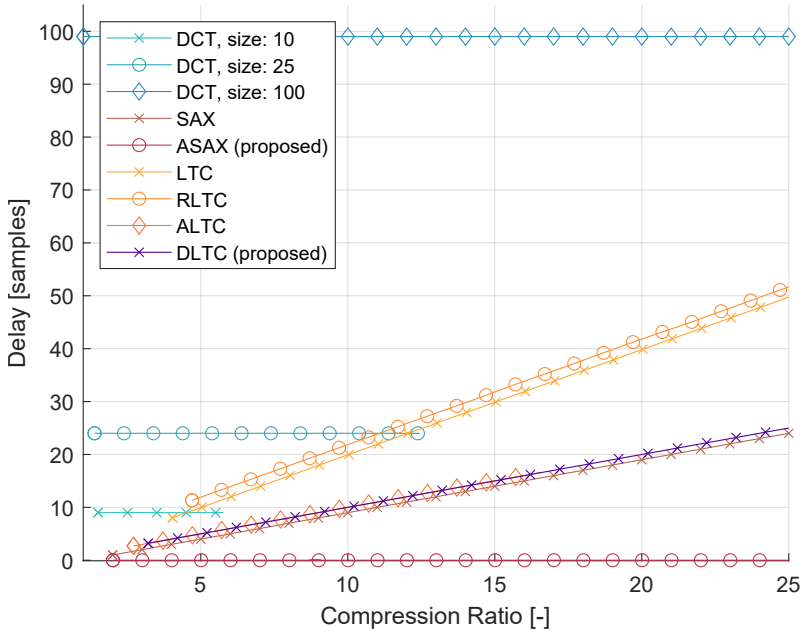


**Figure 2.25** Evaluation of CR and delay trade-off on ECG data

### 2.4.4 Numerical Evaluation Based on Maximum Acceptable RMSE

In the last part of the numerical evaluation, the considered lossy compression methods, namely DCT with RLE (further denoted as DCT), SAX, ASAX, LTC, ALTC, RLTC and DLTC, were evaluated based on the maximum allowed reconstruction error RMSE across the three considered sets of data. The RMSE levels chosen for this evaluation are 1%, 2%, and 5%. These levels were chosen in order to assess the performance of each method at different error thresholds. RMSE higher than 5% is not considered in this evaluation, as the higher deviations impact the data with higher statistical significance.

In the following tables, the remaining performance metrics, namely CR, compression time $\tau_c$, and delay $\nu$ are listed across all methods for RMSE levels of 1%, 2%, and 5% as the maximum allowed reconstruction error. Consequently, the tables include the results of each method for the parameter setting achieving the closest RMSE result below the threshold. In case the data is missing from the table, the method is unable to achieve the error lower than the threshold, or its highest achieved RMSE

is more than 2% lower than the threshold (applicable only in 5% RMSE case).

At all RMSE levels, the most significant performance metric is the CR, which defines the overall effectiveness of the compression technique and its capability to reconstruct the data without damaging it. Compression time $\tau_c$ denotes the complexity of the method, while delay $\nu$ denotes the mean number of samples between each output sample's transmission and the earliest input sample it enables reconstructing.

Table 2.1 contains the results of the evaluation of the heart rate data with the DCT sweep corresponding to the one in Figure 2.17. At 1% RMSE, the highest CR is achieved by DLTC method, with 5.33 CR, 0.32 s dataset compression time, and 5.33 samples of delay. The lowest compression time $\tau_c$ was achieved by SAX and ASAX methods, with considerably poorer CR results. Similarly, DLTC outperforms the other methods in terms of CR at 2% and especially 5% RMSE thresholds, outperforming the other LTC-based methods by a significant margin with more than twice higher CR than RLTC. Additionally, the results show that the delay $\nu$ of DLTC and ALTC methods is equal to their achieved CR, while for LTC and RLTC it is doubled.

The equality of delay and RMSE for DLTC and ALTC is the immediate effect of their algorithm. In case the cut-off is performed at each sample of the sequence, the algorithm reports the last valid sample, and the delay along the whole sequence equals to 1. At the same time, no compression is achieved (CR = 1).

Notice a slight inconsistency between Figure 2.17 and the results in Table 2.1 at maximum RMSE level of 5%, which reports DLTC as a better-performing method. The last valid SAX result was recorded at CR = 50, while for DLTC at CR= 63.5, while the figure considers interpolation between inputs.

The same evaluation, performed at atmospheric pressure data, is presented in Table 2.2 with the corresponding DCT sweep over the window sizes at constant energy 0.99. The atmospheric pressure data represent a series with low dynamics and higher uncertainties within the measurements, as demonstrated in Figure 2.3, in comparison to the heart rate sequence. The results at 1% RMSE show that LTC achieved the highest CR of 4.01. At 2% RMSE threshold, it was outperformed by DLTC with 9.10 CR and favoring delay $\nu$ of only 9.1 samples, whereas at 5% RMSE threshold, DCT outperformed the remaining methods by a large margin, achieving 201.35 CR with the window size of 400 samples.

Table 2.3 shows the evaluation on a set of ECG data, which represents the highly

**Table 2.1** Numerical results on heart rate data

| | RMSE = 1% | | | RMSE = 2% | | | RMSE = 5% | | |
|---|---|---|---|---|---|---|---|---|---|
| Method | CR [-] | $\tau_c$ [s] | $\nu$ [sample] | CR [-] | $\tau_c$ [s] | $\nu$ [sample] | CR [-] | $\tau_c$ [s] | $\nu$ [sample] |
| DCT(10) | 2.92 | 3.79 | 9.00 | 4.90 | 3.93 | 9.00 | - | - | - |
| DCT(25) | 3.32 | 1.17 | 24.00 | 6.73 | 1.16 | 24.00 | 12.50 | 1.16 | 24.00 |
| DCT(100) | 2.79 | 0.29 | 99.00 | 7.24 | 0.26 | 99.00 | 44.97 | 0.24 | 99.00 |
| SAX | 2.00 | 0.15 | 1.00 | 5.00 | 0.06 | 4.00 | 50.02 | 0.01 | 49.00 |
| ASAX | 3.23 | 0.16 | 0.00 | 5.91 | 0.11 | 0.00 | 13.79 | 0.09 | 0.00 |
| LTC | 5.32 | 0.31 | 10.65 | 9.48 | 0.30 | 18.95 | 26.70 | 0.18 | 53.39 |
| ALTC | 2.81 | 0.29 | 2.81 | 5.94 | 0.22 | 5.94 | 19.84 | 0.15 | 19.84 |
| RLTC | 4.97 | 0.88 | 11.94 | 8.56 | 0.65 | 19.12 | 29.71 | 0.39 | 61.42 |
| DLTC | 5.33 | 0.32 | 5.33 | 11.20 | 0.25 | 11.20 | 63.47 | 0.14 | 63.47 |

**Table 2.2** Numerical results on atmospheric pressure data

| | RMSE = 1% | | | RMSE = 2% | | | RMSE = 5% | | |
|---|---|---|---|---|---|---|---|---|---|
| Method | CR [-] | $\tau_c$ [s] | $\nu$ [sample] | CR [-] | $\tau_c$ [s] | $\nu$ [sample] | CR [-] | $\tau_c$ [s] | $\nu$ [sample] |
| DCT | - | - | - | 2.50 | 1.02 | 4.00 | 201.35 | 0.01 | 399.00 |
| SAX | 2.00 | 0.03 | 1.00 | 5.00 | 0.01 | 4.00 | 50.00 | 0.00 | 49.00 |
| ASAX | 3.09 | 0.03 | 0.00 | 4.87 | 0.02 | 0.00 | 11.86 | 0.01 | 0.00 |
| LTC | 4.01 | 0.05 | 8.02 | 7.63 | 0.04 | 15.26 | 26.85 | 0.02 | 53.66 |
| ALTC | 3.02 | 0.05 | 3.03 | 6.91 | 0.03 | 6.92 | 42.45 | 0.01 | 42.57 |
| RLTC | 3.45 | 0.16 | 8.91 | 7.88 | 0.10 | 17.77 | 24.19 | 0.06 | 50.39 |
| DLTC | 3.87 | 0.06 | 3.87 | 9.10 | 0.04 | 9.10 | 135.45 | 0.02 | 135.45 |

dynamic sequence with frequent (linear) spikes and bumps. Due to the aforementioned dynamics, the SAX method is not capable of achieving the reconstruction error lower than RMSE 3.36% at the window size of 2 samples. The best-performing method at all three RMSE thresholds in terms of CR is the novel DLTC, which is capable of maintaining low compression time $\tau_c$ and delay $\nu$.

Based on the results presented above, DLTC is capable of achieving favorable performance in terms of all considered metrics across the different types of wearable-

**Table 2.3** Numerical results on ECG data

| | RMSE = 1% | | | RMSE = 2% | | | RMSE = 5% | | |
|---|---|---|---|---|---|---|---|---|---|
| Method | CR [-] | $\tau_c$ [s] | $\nu$ [sample] | CR [-] | $\tau_c$ [s] | $\nu$ [sample] | CR [-] | $\tau_c$ [s] | $\nu$ [sample] |
| DCT(10) | 2.49 | 0.23 | 9.00 | 3.16 | 0.23 | 9.00 | 4.34 | 0.23 | 9.00 |
| DCT(25) | 2.21 | 0.09 | 24.00 | 2.63 | 0.10 | 24.00 | 4.08 | 0.09 | 24.00 |
| DCT(100) | 1.18 | 0.04 | 99.00 | 1.28 | 0.03 | 99.00 | 1.87 | 0.03 | 99.00 |
| SAX | - | - | - | - | - | - | 2.00 | 0.02 | 1.00 |
| ASAX | 3.20 | 0.01 | 0.00 | 4.72 | 0.01 | 0.00 | 9.07 | 0.01 | 0.00 |
| LTC | 4.04 | 0.04 | 8.04 | 7.83 | 0.03 | 15.60 | 14.53 | 0.03 | 28.91 |
| ALTC | 3.99 | 0.04 | 3.99 | 5.59 | 0.03 | 5.59 | 8.83 | 0.02 | 8.83 |
| RLTC | 4.70 | 0.11 | 11.33 | 8.89 | 0.08 | 19.66 | 15.44 | 0.07 | 32.72 |
| DLTC | 5.17 | 0.03 | 5.17 | 9.33 | 0.03 | 9.33 | 16.97 | 0.03 | 16.97 |

based, time-series data. The ASAX method achieves poorer performance in terms of compression efficiency but successfully minimizes the system latency, represented by the delay metric $\nu$.

## 2.5 Concluding Remarks

In the section above, seven lossy compression methods were evaluated (out of which two were proposed by the Author) on three different wearable-based data sequences, namely a heart rate sequence and atmospheric pressure sequence from a smartwatch, and an ECG sequence, which is obtainable by current smartwatches. Each sequence possesses different trends, uncertainties, and dynamics within the data to evaluate the lossy compression scheme in terms of different applications and scenarios, as all of these methods are highly data-dependent. Among the lossy compression methods, two were developed by the Author.

The evaluation based on the maximum allowed reconstruction error at RMSE levels of 1%, 2%, and 5% show that DLTC is the most robust compression scheme across all RMSE levels and all considered datasets, being outperformed in individual scenarios only by varying methods. The same conclusions can be drawn from the compression performance evaluation, where plotting CR and RMSE trade-off

showed superior stability of DLTC, especially when evaluating the ECG sequence (see Figure 2.19). Evaluating CR and $\tau_c$ trade-off as a compression complexity evaluation shows that SAX achieves the fastest compression across the methods, partially because it is capable of processing the samples in batches. ASAX, ALTC, LTC and DLTC offer fast and lightweight sample-wise compression capabilities. The results visualizing the relation between delay $\nu$ and CR determine that ASAX, as the only of the considered lossy compression techniques, does not create additional system latency to the system, regardless of the CR. The delay of DLTC, ALTC, and SAX methods is increasing linearly with the CR, with the slope of 1. The delay of the DCT method is determined by the DCT window size, and increases linearly with the increasing window size, with a slope of 2. LTC and RLTC methods achieve similar delay-CR ratios due to the offset in the cut-off sequence within the algorithm.

Based on the achieved results and evaluation, the novel DLTC method offers superior compression performance over the other considered lossy compression techniques in terms of wearable computing metrics, with favorable computational complexity in terms of compression time $\tau_c$ and acceptable compression latency in terms of delay $\nu$, which directly corresponds to the achieved CR. Moreover, the implementation of the DLTC method reacts to the sudden changes within data immediately by performing the cut-off at the same instance the trend within data changes, thus longer delays signal linear stability of the considered sequence sector. The recommendation regarding the methods' suitability is summarized in Figure 2.26. ASAX, as the only method among the considered ones, enables delay-free data reduction capabilities with scalable compression performance in terms of CR and RMSE trade-off. SAX is an example of the compression method, that through batch-wise processing achieves the lowest computational strain on the device. For that reason, it is utilized for data mining in the literature [57]. Among all methods, DLTC was found to offer the most consistent performance in terms of complexity, compression performance, and system latency, clearly outperforming the other LTC-based methods.

## 2.6 Author's Contributions

In this chapter, the main contributions to the field of wearable data processing by the Author may be summarized as follows:

- Summarizing the current SotA and pinpointing the applicable existing lossy

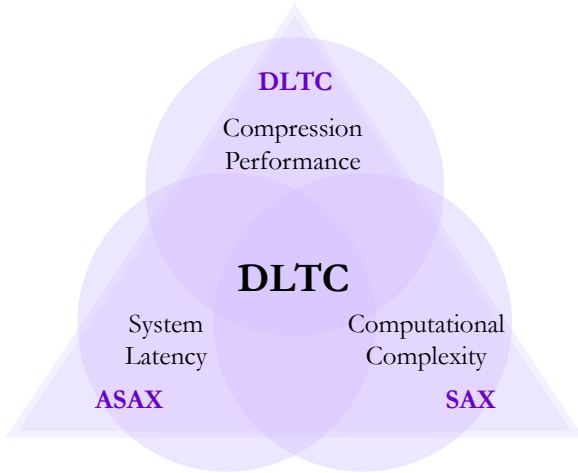


**Figure 2.26** Comparison of methods in terms of fulfilling the three requirements

compression schemes for sensor-based time-series data processing within wearable devices. The algorithm, parameters, advantages, and drawbacks of each considered method are presented.

- Selecting the appropriate performance metrics to evaluate the reconstruction error, compression degree, and algorithmic complexity based on the selected methods and target applications.
- Defining a novel performance metric characterizing the compression mechanism's expected latency, denoted as delay $\nu$.
- Proposing a novel, latency-free Altered Symbolic Aggregate Approximation (ASAX) compression scheme, which enables lightweight and instantaneous compression and reconstruction of the time-series data.
- Proposing a novel Direct Lightweight Temporal Compression (DLTC) compression scheme, optimizing the already efficient LTC method to further boost its performance in terms of minimizing the reconstruction error and algorithmic simplicity. Moreover, DLTC can be effectively used as a lightweight redundancy reduction method since it does not produce arbitrary samples within the output sequence. It also has a smaller delay $\nu$ in comparison to LTC.
- Providing a fair and robust evaluation scheme based on three sets of sensor-based, time-series data, algorithmic implementation of each considered method, and exact definition of obtaining each performance metric.

- Comparatively providing insights into the reconstruction error and compression ratio trade-off of the considered compression schemes, as well as between the compression time and compression ratio trade-off across the considered compression schemes. Additionally, the relation between the achievable CR and delay $\nu$ was addressed.
- Providing numerical results at 1%, 2%, and 5% RMSE threshold of reconstruction error to assess the performance metrics of each method at relevant information loss levels, resulting in a conclusion for the usage of presented methods.
- Recommending the proposed DLTC method as the optimal lossy compression scheme for time-series data in devices with battery restrictions due to its performance, stability, and robustness across different data types.

CHAPTER 3

# DIMENSIONALITY REDUCTION TECHNIQUES FOR EFFORTLESS INDOOR POSITIONING

Ensuring accurate User Equipment (UE) positioning capabilities at all times is one of the main challenges, which Fifth Generation Mobile Networks (5G) and beyond networks need to comply with. In open areas, GNSS technology provides accurate and reliable localization services, yet within indoor scenarios or narrow urban corridors, the signal from satellites is blocked, reflected, and otherwise affected by buildings or other scatterers, leading to unreliable GNSS localization accuracy [71], [72]. In such scenarios, utilizing signals of opportunity, such as cellular, IEEE 802.11 Wireless LAN (Wi-Fi), Ultra Wide-Band (UWB) or Bluetooth Low Energy (BLE) signals can be exploited due to their spatial uniqueness [73]. It is worth mentioning that numerous applications of UWB or BLE infrastructures are implemented as beacons for positioning purposes due to their low energy demands, nevertheless, both technologies can equally serve for communication and connectivity. Such signals

can be described using a number of parameters including signal strength measurements (Received Signal Strength (RSS)), propagation time measurements (Round Trip Time, Time of Flight, Time Difference of Arrival), or spatial measurements (Angle of Arrival, Angle of Departure), each offering various advantages, as well as challenges in their utilization [74], as specified within the current 3GPP standardization [75], [76]. RSS-based positioning offers straight-forward data acquisition, does not require the network nodes and UEs to distinguish the signal's direction of arrival, neither requires the timing synchronization [77], [78].

Although generally in the scope of indoor positioning, the UE is assumed to be a smartphone, any device compatible with the deployed wireless technology standard, including various wearables, IoT devices, tablets, drones, or notebooks can equally be considered. Their network-level presence is overall identical.

Across the existing literature, substantial effort has been made to develop efficient models and algorithms that enable efficient and accurate radio-based positioning, considering either *k*-NN [14], [17], [79], [80] or NN [16], [71], [81] as the positioning models, as well as to determine the necessary parameters of the radio positioning database in terms of coverage and consistency [82]. Both *k*-NN and NN models require a pre-existing dataset of measurements to enable localization in the given scenario, while its quality determines the lower bound for the error of the positioning system - the model can perform only as well as the data it was trained on. Consequently, the positioning system requires plenty of high-quality, location-specific, measurements across the considered deployment, resulting in a voluminous dataset of measured signal strengths across available Access Points (APs) and the corresponding location tags.

In the scope of IoT and wearable devices (to which category smartphones are often considered to belong [10]), the localization services can enable asset and inventory tracking [83], [84], access control and monitoring, and navigation in industrial applications [85], while in eHealth the localization can enable patient tracking [86], movement and fall detection [87], as well as supporting the radio resource management from the network point of view [19], [77]. As the considered *k*-NN fingerprinting positioning system operates within the device, storing and processing the voluminous datasets of measurements to estimate the location requires additional power and resource consumption on devices with limited battery, storage, and performance capabilities. Consequently, introducing lossy compression techniques onto

the radio measurements may lead to strong savings in both the storage and processing requirements of the positioning system. A downside of applying a lossy compression onto the radio map is the possible degradation of the positioning capabilities of the system, which can be regulated, or even avoided by choosing the appropriate dimensionality reduction method and system parameters [44].

The lossy compression methods have been widely used in order to compress the RSS fingerprinting radio maps, including the methods introduced in Section 2.2. For example, the DCT has been used in [51], [52] and achieved significant compression of the input data. At the same time, the positioning performance on the data after the compression has improved in comparison to the positioning on the original radio map. This is seemingly a result of the DCT's ability to suppress the high frequencies in the data and thus reduce the noise present in the original measurements. However, the downside of the applicability of DCT on radio maps lies in the necessity of applying iDCT in order to recover the radio map and perform the positioning. Updating the radio map with the new samples is also challenging.

SAX has also been used in terms of Radio Frequency (RF) fingerprinting in [88], where it was applied as a time-series compression used on the bursts of RF emissions using IoT devices. The method showed promise in terms of both accuracy and reduction of processing time.

### Compression Mechanisms for Positioning

The lossy compression techniques introduced in this chapter are designed to reduce the size of the radio map used for positioning purposes. As the radio measurements are stored as a radio map in a single matrix, the radio map compression schemes can operate in three different dimensions, as depicted in Figure 3.1. The bit-level compression denotes the reduction of every single element of the radio map to a lower-bit representation. Feature-wise compression reduces the number of elements, which represent every fingerprint, while sample-wise compression minimizes the number of samples themselves.

The general positioning scheme with an arbitrary compression method and $k$-NN localization algorithm is represented in Figure 3.2. The compression mechanism within the positioning scheme is implemented before the positioning model onto a radio map, comprised of the training features. The training labels (coordinates, floor index, building index, etc.) are not affected by the compression mechanism.

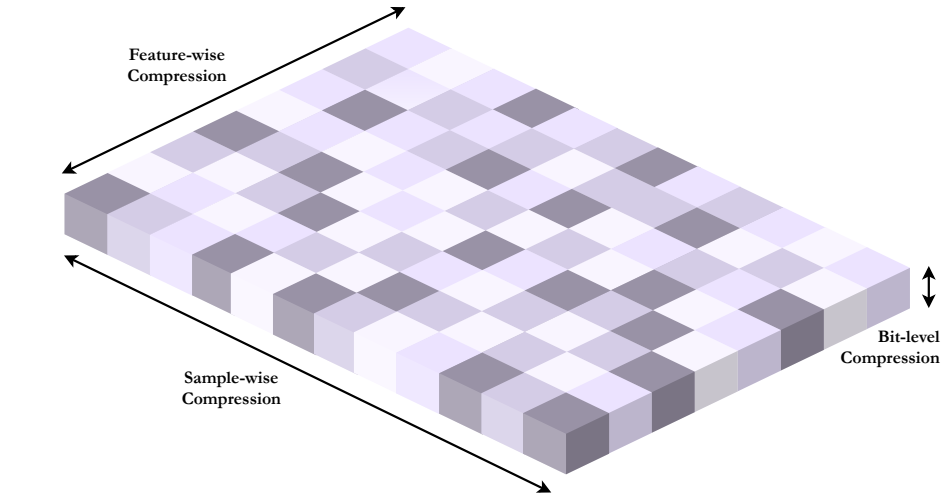


**Figure 3.1** Considered radio map compression dimensions

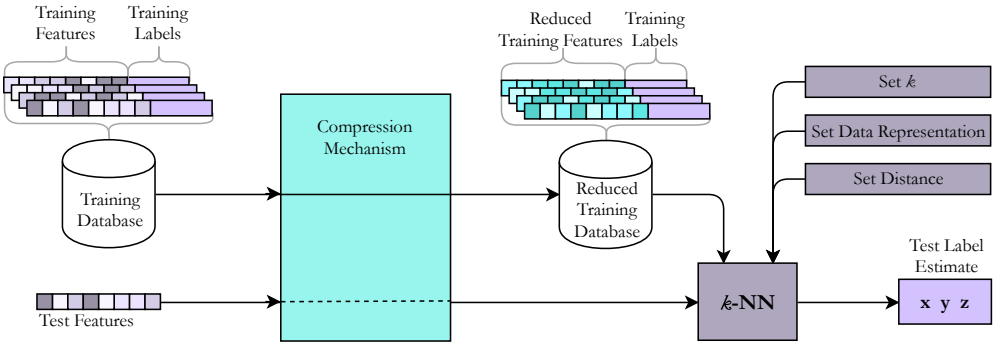


**Figure 3.2** Compression mechanism for indoor positioning

After the lossy compression is applied, the training database can be stored in its reduced representation, while saving storage resources. The test features, or the measured RSS array that needs to be localized, are first compressed using the same mechanism to ensure equal representation as the training data, and then localized using the positioning model. If not stated otherwise, the compression schemes in this thesis are applied in a way consistent with the algorithm in Figure 3.2.

The tunable parameters of the $k$-NN model are the number of neighbors $k$, the distance metric that is used to calculate similarities between the samples, and the function transforming the RSS measurements, originally obtained in dBm, to an altered representation. The description of $k$-NN is included in the paragraphs below.

# 3.1 Positioning Basics, Performance Metrics, Utilized Data, and Baselines

In this section, the relevant performance metrics in the scope of positioning and compression co-existence are explained and later used in order to pinpoint the main goals and contributions of this chapter (see Section 3.1.2). To demonstrate them and to evaluate the proposed methods, 26 different datasets are used. They are introduced in Section 3.1.4, while the baselines to which the performance is compared are introduced in Section 3.1.5. Moreover, Section 3.1.3 introduces a novel dataset measured and processed by the Author, which focuses on addressing the challenge of device diversity within indoor positioning systems.

All evaluations within this chapter were performed using MATLAB 2021*b* on a computer with 32 GB of RAM, and an Intel Core i7-8750H CPU besides those involving time measurements. Those were evaluated using MATLAB Online for performance consistency.

## 3.1.1 Positioning Basics

Before introducing the applicable compression mechanisms, the considered positioning scheme is first described to assess the behavior and the benchmark performance of the system without compression. This section first introduces the *k*-NN as a positioning mechanism, data representations of the radio map, and the distance (similarity) metrics utilized within this thesis.

### *k*-Nearest Neighbors (*k*-NN)

*k*-NN is a simple, yet powerful matching algorithm with numerous applications across current literature. Despite the existence of powerful DL models, its performance is often unmatched, especially in situations when the amount of available data is limited.

In the scope of indoor positioning, *k*-NN's popularity and robustness in hardly ever outmatched due to dataset limitations, level of uncertainties within the measurements, as well as the fact that rather than learning on labels, as is the case in e.g. NNs, *k*-NN performs the matching based purely on the similarity of features. Just

like ML methods, $k$-NN requires a pre-existing and labeled database, using which the matching is realized.

$k$-NN compares the current sample to all available samples in the (training) database and distinguishes $k$ samples with the highest similarity measure, called a distance metric. In this thesis, 9 distance metrics are considered [89]. The labels of the $k$ "neighbors" are used to estimate the labels of the currently considered sample. The $k$-NN algorithm can be implemented as either regressor or classifier ($k = 1$ or more with majority voting), depending on the label processing algorithm. In the scope of indoor positioning, estimating the building and floor is considered a classification task, while obtaining the exact user coordinate represents a regression.

The main disadvantages of $k$-NN include the necessity to obtain the similarity of the evaluated sample to all training samples, leading to lengthy computations, especially if voluminous datasets are considered. Consequently, $k$-NN's positioning performance generally increases with a larger training database, while its computational strain on the system increases as well. $k$-NN is further vulnerable to outliers and erroneous samples within the training database, which may lead to misclassifications. Additional challenges include choosing a suitable value of $k$ and finding the best-performing parameters in terms of distance calculation.

In the scope of this thesis, $k$-NN is applied to a training radio map of RSS measurements with the corresponding $x$, $y$, and $z$ coordinates, as well as the building and floor labels. Before applying the $k$-NN, a data representation transformation is applied to the database to further boost the positioning capabilities.

### Considered Data Representations

In this thesis, three data representations are considered. Their implementation is based on [90], namely *positive* data representation, *exponential* data representation with $\alpha = 24$, and *powed* data representation with $\beta = e$ are considered, where $\alpha$ and $\beta$ refer to the specific constants in the corresponding formulas introduced in e.g. [90].

The *positive* data representation shifts the raw RSS measurements to the positive values by subtracting the minimum measured RSS value from all feature elements. The *exponential* data representation applies the exponential function to all elements after applying positive representation. Finally, *powed* representation rises the elements in positive representation to the $\beta^{th}$ power.

Notably, *positive* data representation is a linear transformation that shows the

magnitude of the measured signal above the considered floor, while *exponential* and *powed* representations apply an additional non-linearity, both amplifying the stronger measurements more than the poor ones.

### Considered Distances

The formula for calculating the degree of similarity, denoted distance metric, is the critical parameter of the $k$-NN, defining its overall performance. Across SotA, there are numerous different distance metrics, as summarized in [89]–[92], from which Manhattan (Cityblock or L1) distance, Euclidean (L2) distance, Squared Euclidean distance, Hamming distance, Logarithmic Gaussian distance (LGD), Neyman distance, Penalized Logarithmic Gaussian distance with penalty 10 (PLGD10) and 40 (PLGD40), and Sørensen distance are considered, based on the recent literature [79], author's prior experience, and distance characteristics.

This work considers Sørensen distance as the reference one, applying it in $\alpha$ baseline, due to its simplicity and superior performance capabilities compared to the traditional Manhattan or Euclidean distances [79]. E.g. in [21], Sørensen achieved approximately 10% better positioning capabilities on the considered datasets (many of them also utilized in this thesis). On the other hand, according to the same evaluation, the execution time of Sørensen distance is higher by approx. 16% in comparison to Manhattan distance. Nonetheless, Sørensen distance is considered a better fitting choice for the upcoming evaluation since positioning performance is considered a more important metric in this thesis. Additionally, the techniques of how to decrease the positioning time are discussed later in Section 4.1. The Sørensen distance (denoted as $dist_{Sorensen}$) is obtained as:

$$dist_{Sorensen}(f_{\mathbf{i}}, \overline{f}_{\mathbf{i}}) = \frac{\sum_{a=1}^{|AP|} |f_{i,a} - \overline{f}_{i,a}|}{\sum_{a=1}^{|AP|} f_{i,a} + \overline{f}_{i,a}} \tag{3.1}$$

where $f_{\mathbf{i}}$ and $\overline{f}_{\mathbf{i}}$ denote the $i^{th}$ sample's RSS feature vector and its estimate, respectively, while $f_{i,a}$ and $\overline{f}_{i,a}$ denote their $a^{th}$ element's numerical value. For exact expressions and algorithms for the remaining metrics please refer to the aforementioned referred literature.

### 3.1.2 Performance Metrics

In this section, the performance metrics considered for the evaluation of positioning-related capabilities of proposed methods are introduced.

#### Building Hit - $\zeta_{\mathcal{B}}$

The building hit $\zeta_{\mathcal{B}}$ represents the capability of the system to detect the correct building $\mathcal{B}$ across the testing samples. It is expressed in percentages. Although the majority of the available datasets are measured only in one building, the datasets UJI 1 and UJI 2 both span over three buildings. The ability to correctly detect the building is the most impactful capability of the positioning system, as each misclassification of a building label results in serious performance degeneration. The building hit $\zeta_{\mathcal{B}}$ is obtained as:

$$\zeta_{\mathcal{B}} = \frac{1}{|\mathbf{S_{test}}|} \sum_{i=1}^{|\mathbf{S_{test}}|} (\mathcal{B}_i == \overline{\mathcal{B}}_i) \cdot 100\% \tag{3.2}$$

where $|\mathbf{S_{test}}|$ represents the number of samples in the testing dataset, $\mathcal{B}_i$ denotes the true building label of the $i^{th}$ sample and $\overline{\mathcal{B}}_i$ denotes the estimated building index of the $i^{th}$ sample, while the operator == stands for element-wise equal-to operator.

#### Floor Hit - $\zeta_{\mathcal{F}}$

The floor hit $\zeta_{\mathcal{F}}$ evaluates the ability of the system to correctly detect the given floor $\mathcal{F}$ in the given building $\mathcal{B}$. Similarly to building hit, it is expressed in percentages. Most of the datasets used for the evaluation in this thesis span over two and more floors. Consequently, the floor hit is considered to be the second most important metric (after building hit) as the capability of the system to correctly detect the floor is in terms of user navigation much more important than the actual 2-Dimensional (2D) positioning error. The floor hit $\zeta_{\mathcal{F}}$ is defined as:

$$\zeta_{\mathcal{F}} = \frac{1}{|\mathbf{S_{test}}|} \sum_{i=1}^{|\mathbf{S_{test}}|} (\mathcal{B}_i == \overline{\mathcal{B}}_i \ \ \& \ \ \mathcal{F}_i == \overline{\mathcal{F}}_i) \cdot 100\% \tag{3.3}$$

where $\mathcal{F}_i$ denotes the true floor label of the $i^{th}$ sample and $\overline{\mathcal{F}}_i$ denotes the estimated

floor index of the $i^{th}$ sample. The & denotes element-wise logical "AND".

### 3D and 2D Positioning Error - $\varepsilon_{3D}$, $\varepsilon_{2D}$

The 3D positioning error $\varepsilon_{3D}$ represents the error of the position estimates to the original labels. It is calculated as the mean Euclidean distance between the location labels of the training dataset $\mathbf{S_{train}}$ and the positioning estimates in $x$, $y$, and $z$ dimensions. Formally, the 3D positioning error $\varepsilon_{3D}$ is obtained as:

$$\varepsilon_{3D} = \frac{1}{|\mathbf{S_{test}}|} \sum_{i=1}^{|\mathbf{S_{test}}|} \sqrt{\sum_{d=1}^{3} \left(l_{i,d} - \bar{l}_{i,d}\right)^2} \tag{3.4}$$

where $l_{i,d}$ denotes $d^{th}$ element of the $i^{th}$ sample's label, and $\bar{l}_{i,d}$ denotes $d^{th}$ element of the $i^{th}$ sample's estimated label.

The 3D positioning error $\varepsilon_{3D}$ expresses the cumulative error of the building hit, floor hit, and the local positioning error, and is generally considered as the target optimization metric (e.g. in [79] the best-performing baseline was based on $\varepsilon_{3D}$).

The 2D positioning error $\varepsilon_{2D}$ characterizes the local positioning error, while not considering the misclassified samples in terms of building and floor hit in the calculation. It can be interpreted as the expected positioning error, given the known general location. 2D positioning error $\varepsilon_{2D}$ can be expressed as:

$$\varepsilon_{2D} = \frac{100}{|\mathbf{S_{test}}| \cdot \zeta_{\mathcal{F}}} \sum_{i=1}^{|\mathbf{S_{test}}|} \sqrt{\sum_{d=1}^{2} \left(l_{i,d} - \bar{l}_{i,d}\right)^2} \cdot (\mathcal{B}_i == \overline{\mathcal{B}}_i \ \ \& \ \ \mathcal{F}_i == \overline{\mathcal{F}}_i) \tag{3.5}$$

Consequently, $\varepsilon_{3D}$ is identical to $\varepsilon_{2D}$ in the single-floor datasets, or when floor hit $\zeta_{\mathcal{F}} = 100\%$. The normalization using the floor hit ensures the correct scaling while discarding the misclassified samples.

### Time of Prediction - $\tau$

The time of prediction $\tau$ represents the effort required by the positioning system to acquire a positioning estimate of a sample. It is obtained as a mean elapsed time, by estimating the position of the test sample. Time of prediction $\tau$ may be affected by the size of the training dataset - e.g. the prediction time of $k$-NN model changes based

on the dataset size (number of training samples) considerably. This is caused by the $k$-NN method itself since it requires a calculation of the distance from the considered sample to all samples in the training dataset. Consequently the bigger the dataset is the higher number of operations required. In contrast, this is not the case for e.g. NN applications, where, after a considerably longer training time, the localization of the user using a pre-trained network always takes the same time, regardless of the training dataset size. Other aspects, such as a number of APs, affect the required effort regardless of the model.

The training time $\tau_{train}$, which represents the time required for the training phase of the method is not considered as important as the time of prediction in this thesis, mainly because it is assumed that the positioning system may be pre-trained before it is being implemented, and thus it does not concern the user or the actual UE (such as wearable). Notably, there is no training time required for $k$-NN, while NNs require substantially more time to train. This time is mainly based on the complexity of NN architecture, used optimizer, loss, etc. For the consistency of the results, MATLAB Online was used for the evaluation concerning the time estimates.

### Normalization to baseline

In order to fairly evaluate the proposed methods and to highlight the different performances they achieve, the normalization to two baselines is used. Each of the two baselines (denoted as baseline $\alpha$ and baseline $\beta$) is introduced in the following part of this section, including their numerical evaluation for each of the considered datasets. To differentiate the considered metric (e.g. a dummy metric $\Gamma$) from the normalized values to their given baseline, a tilde is used above the metric's symbol (e.g. $\tilde{\Gamma}$). Furthermore, in order to distinguish the baselines apart, a subscript of $\alpha$ or $\beta$ is used next to the standard symbol (e.g. $\tilde{\Gamma}_{\alpha}$).

The normalization is obtained as $\tilde{\Gamma}$ in Equation 3.6 for each of the considered metrics ($\zeta_{\mathcal{B}}$, $\zeta_{\mathcal{F}}$, $\varepsilon_{3D}$, $\varepsilon_{2D}$, $\tau$), where $\Gamma_{test}$ represents the evaluation metric value after applying the tested methods, and $\Gamma_{baseline}$ represents the evaluation metric value obtained after applying the considered baseline.

$$\tilde{\Gamma} = \frac{\Gamma_{test}}{\Gamma_{baseline}} \tag{3.6}$$

Normalizing the metrics allows for a fair comparison of the utilized method

across many datasets, as it expresses the ratio of metric improvement or degradation. Furthermore, it allows for efficient metric aggregation, as introduced in [21], which enables describing the methods' performance across multiple datasets with a single expression.

In terms of metrics introduced in this chapter, the normalized building or floor hit larger than 1 means the performance has improved, as a higher floor/building hit is preferred. In contrast with this, the normalized 2D or 3-Dimensional (3D) positioning error larger than 1 means the performance has degraded, as a lower positioning error is preferred.

### Compression Ratio - CR

As already introduced in Section 2.1, CR represents the multitude of the size reduction achieved by applying the given dimensionality reduction method. In the scope of this thesis, numerous types of dimensionality reduction methods are applied onto the radio map. Further in this work, a combined compression scheme is introduced and evaluated as well. In general, CR is obtained as introduced in Equation 2.1, regardless of the compressed dimension. The methods in this thesis perform a bit-level compression, where each element of the radio map is brought to a lower-bit representation, feature-level compression, where the number of features (APs) is reduced or transformed, or sample-wise compression, where the total number of samples in the training dataset is reduced.

Specifically, a bit-level CR is obtained by comparing the size reduction of a single element of the radio map in terms of bits. Since the compression is the same for all elements in the whole dataset, it is not necessary to consider the size of the whole dataset. Later in this thesis, a 7-bit baseline is introduced and utilized, but for the sake of completeness, a more universal expression is used:

$$CR_{bit} = \frac{(number\ of\ bits\ in\ original\ format)}{(number\ of\ bits\ after\ compression)} \tag{3.7}$$

The feature-wise and sample-wise CR is calculated as the ratio of the original radio map size to the reduced radio map size in the given dimension.

The combined CR, denoted $CR_{tot}$ is calculated as the product of the individual CRs of the methods utilized in the series, given that each individual compression $CR_n$ is applied in different dimension:

$$CR_{tot} = \prod_{n=1}^{N} CR_n \tag{3.8}$$

where $n$ denotes the index of a compression dimension and $N$ is the total number of compression mechanisms applied in a series.

### Root Mean Squared Error - RMSE

RMSE is used to express the reconstruction error resulting from applying a lossy compression on data, as introduced in Section 2.1. The calculation follows Equation 2.2 for RMSE in absolute units, while substituting the vectors of $y$-axis values for the radio map elements as:

$$RMSE = \sqrt{\frac{1}{|\mathbf{S_{test}}| \cdot |AP|} \sum_{s=1}^{|\mathbf{S_{test}}|} \sum_{a=1}^{|AP|} [f_{s,a} - \overline{f}_{s,a}]^2} \tag{3.9}$$

where $f_{s,a}$ denotes the measured RSS of the $s^{th}$ sample at $a^{th}$ AP and $\overline{f}_{s,a}$ denotes the corresponding element after the radio map compression.

The difference in RMSE achieved by two different schemes is denoted as:

$$\eth_{RMSE} = RMSE_{S1} - RMSE_{S2} \tag{3.10}$$

where $RMSE_{S1}$ denotes RMSE of the method $S1$ and $RMSE_{S2}$ denoted RMSE of the method $S2$.

## 3.1.3 TUJI 1 Dataset (collected by the Author)

Device heterogeneity presents one of the leading challenges in the area of indoor positioning. Nowadays, the market with Wi-Fi-enabled devices ranges from hundreds of cellular phones, through tablets, notebooks, IoT devices and wearables, where every model's physical (antenna, amplifiers, etc.) and software (Wi-Fi standard compliance, etc.) parameters differ. Such diversity leads to varying signal measurements, despite constant environmental and RF conditions. To demonstrate, address, and consider such challenges, a multi-device indoor positioning dataset was created.

The dataset denoted TUJI 1 was collected, processed, and evaluated by the Author

during her doctoral studies. The dataset was measured at the premises of Universidad Jaume I., Castellón de la Plana (UJI), Spain, and is a result of a collaboration between UJI and Tampere University (TAU), Finland. In total, 5 devices were used. The considered entities utilized as features are Wi-Fi RSS measurements, despite more entities being collected during the site survey, such as BLE RSS, acceleration, physical orientation, or GNSS signals. Each sample was collected at an accurately pre-defined and labeled location, ensuring high labeling accuracy.

The considered scenario was a set of offices on the university grounds and the open space between them. In total, almost $9,000$ samples were collected and labeled using the mobile application GetSensorData [93], version 2.1. The basic information regarding the dataset can be found in Table 3.1, while the device-specific information is included in Table 3.2.

**Table 3.1** TUJI 1 dataset information

| | | | | | |
|---|---|---|---|---|---|
| $\lvert\mathbf{S}\rvert$ | 8899 | $\lvert AP\rvert$ | 310 | $\lvert\mathcal{B}\rvert$ | 1 |
| $\lvert\mathbf{S_{train}}\rvert$ | 6752 | $\mu_{S_{train}loc}$ | 5 | $\lvert\mathcal{F}\rvert$ | 1 |
| $\lvert\mathbf{S_{test}}\rvert$ | 2147 | technology | Wi-Fi | format | int |

**Table 3.2** TUJI 1 dataset device-specific information

| Device | Device type | $\lvert\mathbf{S}\rvert$ | $\lvert\mathbf{S_{train}}\rvert$ | $\lvert\mathbf{S_{test}}\rvert$ | $\lvert AP\rvert$ | OS |
|---|---|---|---|---|---|---|
| Galaxy *S*20 | Phone | 1828 | 1408 | 420 | 244 | Android 11 |
| Galaxy *S*7 | Phone | 1756 | 1336 | 420 | 180 | Android 8 |
| Galaxy Tab *S*7+ | Tablet | 1747 | 1318 | 429 | 229 | Android 11 |
| Galaxy *A*12 | Phone | 1835 | 1377 | 458 | 120 | Android 11 |
| POCO | Phone | 1733 | 1313 | 420 | 204 | Android 11 |

To represent the device heterogeneity, the selected devices include an off-the-shelf flagship phone Samsung Galaxy S20, two mid-range phones Xiaomi POCO X3 and Samsung Galaxy A12, as well as an older flagship model Samsung Galaxy S7. To increase the device diversity, a tablet Samsung Galaxy Tab S7+ was added to the list of devices. Other devices, such as notebooks or smartwatches can equally be utilized in such surveys as they comply with the same standards.

Each device fully surveyed the floor plan prepared for the evaluation. The mea-

surements were taken in two rectangular grids, each 60 cm by 60 cm, intersecting each other (so that each measurement from the second grid was in the center of the first grid's rectangle). The two grids were processed separately, then half of the locations from the second grid was designated for testing, along with all samples carrying their labels. No measurements with the same label belong to both testing and training datasets. The visualization of the split between training and testing samples is shown in Figure 3.3, which also captures the geometry of the deployment.

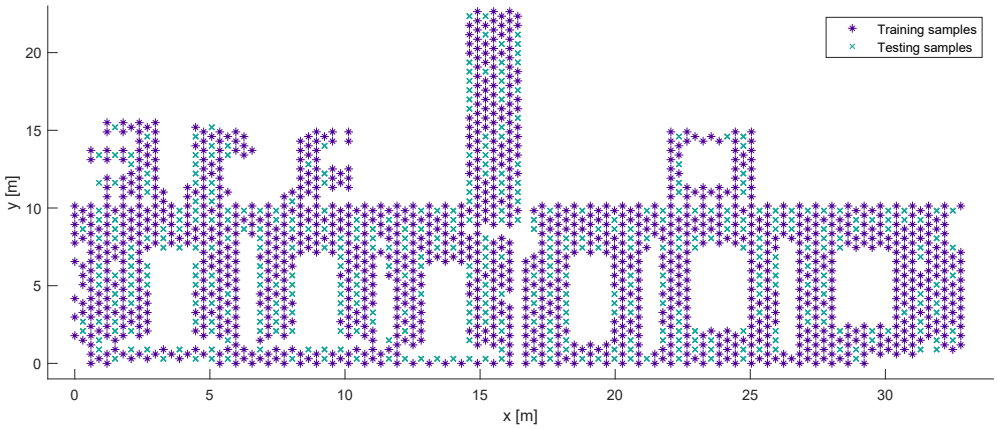


**Figure 3.3** Map of testing and training samples for TUJI1 dataset

To demonstrate the device heterogeneity, the Cumulative Distribution Function (CDF), as well as the histogram of the RSS measurements within the whole dataset are provided in Figure 3.4. The CDF demonstrates the shape of the distributions and the visible shift is noticeable when comparing the individual devices, especially Samsung S20, whose distribution is visibly shifted to the left by approx. 8 dB when compared to the tablet. To also capture the measurement counts, the histogram figure (right) provides the quantitative comparison of the measured values, where the statistics for Samsung S20 show remarkably more measurements between −95 dBm and −85 dBm than those of the remaining devices, signifying higher receiver sensitivity than that of the remaining devices. Notably, at the higher signal strength values, the counts across most of the devices are similar. The histogram of Samsung A12 shows considerably lower measurement counts than that of the remaining devices, which is attributed to its Wi-Fi module limitations to 802.11 b/g/n standards at the 2.4 GHz band.

To assess the correctness of the pre-processing and matching process after the

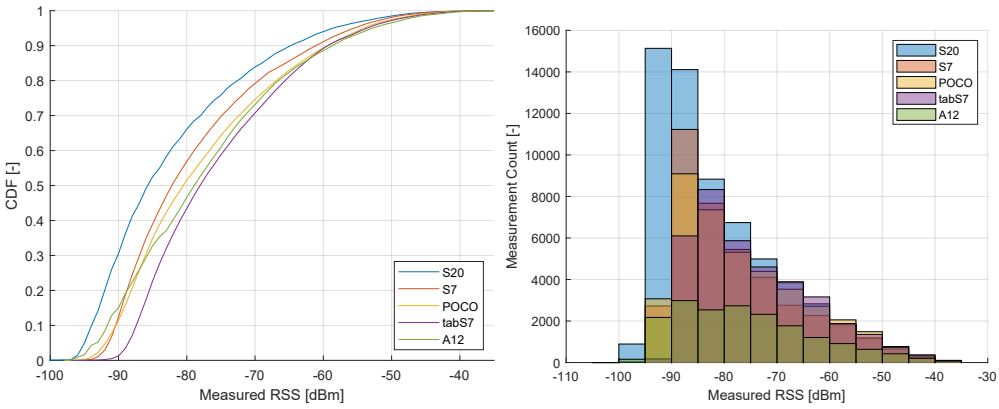


**Figure 3.4** The CDF (left) and the histogram (right) of the RSS measurements per utilized device

measurements were merged, the visualization of the data was performed. Figure 3.5 shows the signal strength distribution across all available samples from the AP located on the ceiling of the open-space office on the left side of the figure. The consistent signal propagation patterns can be observed in the figure, indicating no mismatched labels.

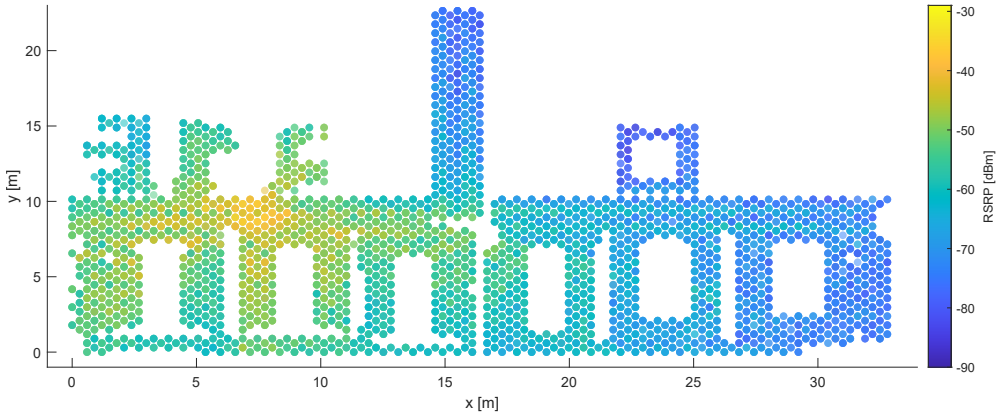


**Figure 3.5** Example 1 of a signal strength distribution for a single AP in TUJI1 dataset

The second AP's radio map is shown in Figure 3.6 to visually assess the correctness of processing the data on the other side of the deployment. The AP's location is in a small office on the right side of the considered scenario. Notably, in the small offices on the left side of the deployment the AP is not detected anymore (see the

missing points in the radio map, comparing Figures 3.5 and 3.6).

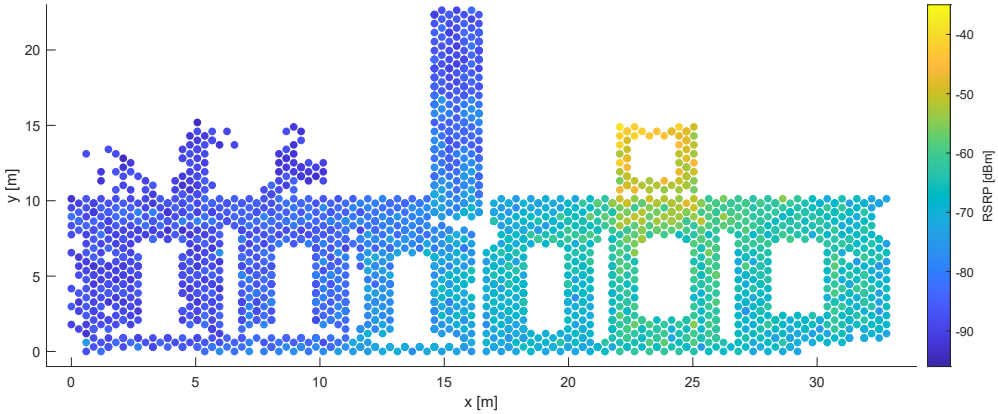


**Figure 3.6** Example 2 a of signal strength distribution for a single AP in TUJI1 dataset

The preliminary positioning performance achieved on the created dataset was assessed by applying the $k$-NN model to the data with $\beta$ baseline settings, as described later in the text, with the number of considered neighbors $k = 7$, Sørensen distance metric and exponential data representation. Figure 3.7 illustrates the achieved positioning errors at the training locations via heat map, as well as visualizes the estimated testing locations (black stars).

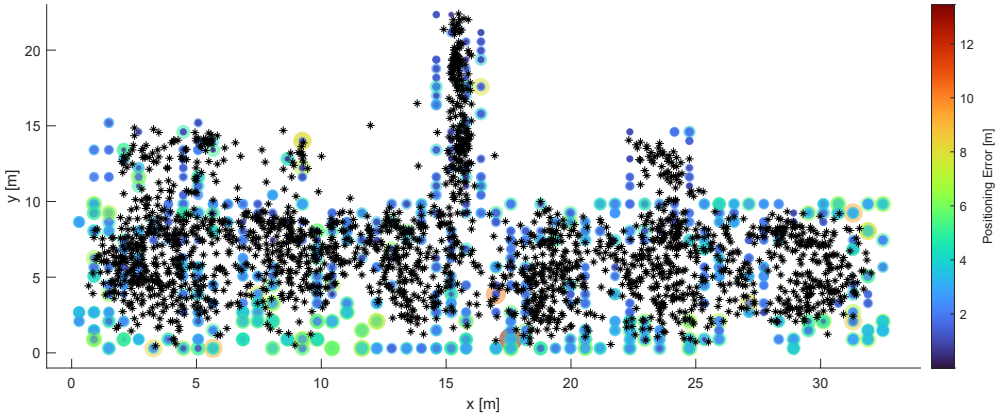


**Figure 3.7** Heat map of $k$-NN estimation errors and estimated testing locations (marked by black stars)

The results show the expected behavior of the $k$-NN estimator, pulling the individual estimates towards the center of the room (especially visible in the corridor in

the middle). The figure shows that the mean positioning error of 2.27 m indicates the good capability of the positioning model to localize the user on a room-level, but pinpointing the accurate device location is not always feasible in indoor positioning scenarios utilizing Wi-Fi RSS signals. As shown in Figures 3.5 and 3.6, the signal strength from a single AP is spread throughout the considered environment, limiting the accuracy of the localization system by the number of unique and distinguishable APs. Moreover, the evaluation shows that uncertainties within the measurements caused by signal fluctuations, device heterogeneity, orientation, as well as other environmental effects cause the positioning performance to be significantly lower than the grid resolution.

The numerical and visual evaluations of the collected dataset indicate consistent trends in terms of signal propagation as well as classifier behavior across the considered deployment. In the sections below, the dataset is employed as one of the 26 indoor positioning scenarios utilized for evaluating the proposed radio map compression schemes.

### 3.1.4 Considered Indoor Positioning Datasets

The methods discussed in this and the following section are developed mainly for fingerprinting-based positioning, even though their applicability in other fields is possible. In the scope of positioning, the evaluation of their performance is based on seven key performance metrics, demonstrated by their application onto (up to) 26 different datasets. These datasets (or their subset) were used across the Author's previous positioning-related papers, as well as in many other works released within the indoor positioning community. The datasets are collected at different locations, with various technologies, and vary in the number of samples. Evaluating each method using many heterogeneous datasets provides robust and fair performance comparison and allows for drawing unbiased conclusions regarding each method's strengths and weaknesses.

All considered datasets are openly available online, apart from dataset TUJI 1 (collected by the author of this thesis), which shall be available online soon. For the purposes of smooth evaluation, the datasets are in the form of a consistent MATLAB format structure: each dataset consists of four matrices, representing a set of training features, a set of training labels, a set of testing features, and a set of testing labels.

The set of training features is represented by a $|\mathbf{S_{train}}|$ by $|AP|$ matrix of individual radio measurements, containing all training samples. The set of testing features is a $|\mathbf{S_{test}}|$ by $|AP|$ matrix of individual radio measurements of the testing samples, where $f_{s,a}$ denotes the measured RSS feature of the $s^{th}$ sample at $a^{th}$ AP.

The set of training labels is a $|\mathbf{S_{train}}| \times 5$ matrix of training samples' labels. Similarly, the set of testing labels is a $|\mathbf{S_{test}}| \times 5$ matrix of labels belonging to the testing samples, where the individual labels are denoted as $l_{i,d}$ for $d^{th}$ label of $i^{th}$ sample. The 5 columns denote $x$-axis coordinate (for $d = 1$), $y$-axis coordinate (for $d = 2$), $z$-axis coordinate (for $d = 3$), floor label, and building label, respectively.

The most frequently used data collection technology across considered datasets is Wi-Fi, followed by BLE datasets and one simulated dataset. The dataset sizes vary from low-sample datasets with approx. 300 samples to bigger-size datasets with over 26000 samples. Although the dataset sizes may be insufficient for training complex ML algorithms, such as DL models, they are sufficient for the demonstration of the proposed compression and localization methods.

Generally, the datasets using BLE as a wireless technology are smaller in size (the biggest one consists of 1632 samples), whereas the Wi-Fi datasets range from 760 to 26151 samples. The simulated dataset includes 11710 samples in total. Several proposed methods are not evaluated on the full set of datasets, omitting the ones with an insufficient number of training samples.

The datasets were collected within various environments, at six different universities spread across Europe and Australia. The universities include Tampere University, Finland (TAU), University Jaume I, Spain (UJI), University of Minho, Portugal (UMi), University of Extremadura, Spain (UEX), University of Mannheim, Germany (UMa) and University of Sydney, Australia (USYD). The deployment environments covered by the datasets range from one (in most cases) to three distinct buildings, and from one to 16 floors. Similarly, some of the datasets do not consist of multiple measurements per point, which is a key requirement for the methods described in this section and therefore could not be used for their evaluation.

More detailed information regarding each of the datasets, including their corresponding reference, may be found in Table 3.3. The table specifies the total number of samples $|\mathbf{S}|$, training a testing set size ($|\mathbf{S_{train}}|$ and $|\mathbf{S_{test}}|$), training sample density per location $\mu_{S_{train}loc}$, number of APs ($|AP|$), buildings $|\mathcal{B}|$, and the total number of floors $|\mathcal{F}|$, as well the sample format (float or integer, which is denoted as int). A

**Table 3.3** Basic dataset information

| Dataset | $\|\mathbf{S}\|$ | $\|\mathbf{S}_{\mathbf{train}}\|$ | $\|\mathbf{S}_{\mathbf{test}}\|$ | $\mu_{S_{train}loc}$ | $\|AP\|$ | $\|\mathcal{B}\|$ | $\|\mathcal{F}\|$ | Technology | Format | Provided by | Reference |
|---|---|---|---|---|---|---|---|---|---|---|---|
| DSI 1 | 1717 | 1369 | 348 | 5.95 | 157 | 1 | 1 | Wi-Fi | int | UMi | [94] |
| DSI 2 | 924 | 576 | 348 | 2.5 | 157 | 1 | 1 | Wi-Fi | int | UMi | [94] |
| LIB 1 | 3696 | 576 | 3120 | 12 | 174 | 1 | 2 | Wi-Fi | int | UJI | [95] |
| LIB 2 | 3696 | 576 | 3120 | 12 | 197 | 1 | 2 | Wi-Fi | int | UJI | [95] |
| MAN 1 | 14760 | 14300 | 460 | 110 | 28 | 1 | 1 | Wi-Fi | int | UMa | [96], [97] |
| MAN 2 | 1760 | 1300 | 460 | 10 | 28 | 1 | 1 | Wi-Fi | float | UMa | [96], [97] |
| MINT 1 | 5783 | 4973 | 810 | 26.31 | 11 | 1 | 1 | Wi-Fi | float | UMi | [98] |
| SAH 1 | 9447 | 9291 | 156 | 1 | 775 | 1 | 3 | Wi-Fi | int | TAU | [99] |
| SIM 1 | 11710 | 10710 | 1000 | 10 | 8 | 1 | 1 | simulated | int | UJI | [79] |
| TIE 1 | 10683 | 10633 | 50 | 1 | 613 | 1 | 6 | Wi-Fi | int | TAU | [99] |
| TUJI 1 | 8899 | 6752 | 2147 | 5 | 310 | 1 | 1 | Wi-Fi | int | TAU/UJI | * |
| TUT 1 | 1966 | 1476 | 490 | 1 | 309 | 1 | 4 | Wi-Fi | float | TAU | [100], [101] |
| TUT 2 | 760 | 584 | 176 | 1 | 354 | 1 | 3 | Wi-Fi | float | TAU | [100], [101] |
| TUT 3 | 4648 | 697 | 3951 | 1 | 992 | 1 | 5 | Wi-Fi | int | TAU | [102] |
| TUT 4 | 4648 | 3951 | 697 | 1.03 | 992 | 1 | 5 | Wi-Fi | int | TAU | [102] |
| TUT 5 | 1428 | 446 | 982 | 1 | 489 | 1 | 3 | Wi-Fi | float | TAU | [103] |
| TUT 6 | 10385 | 3116 | 7269 | 1 | 652 | 1 | 4 | Wi-Fi | int | TAU | [104] |
| TUT 7 | 9291 | 2787 | 6504 | 1 | 801 | 1 | 3 | Wi-Fi | int | TAU | [104] |
| UEX B1 | 519 | 417 | 102 | 3 | 30 | 1 | 4 | BLE | float | UEX | [105] |
| UEX B2 | 690 | 552 | 138 | 3 | 30 | 1 | 5 | BLE | float | UEX | [105] |
| UEX B3 | 300 | 240 | 60 | 2 | 30 | 1 | 5 | BLE | float | UEX | [105] |
| UJI 1 | 20972 | 19861 | 1111 | 21.29 | 520 | 3 | 13 | Wi-Fi | int | UJI | [106] |
| UJI 2 | 26151 | 20972 | 5179 | 10.51 | 520 | 3 | 13 | Wi-Fi | int | UJI | [106] |
| UJI B1 | 1632 | 732 | 900 | 30.5 | 24 | 1 | 1 | BLE | float | UJI | [107] |
| UJI B2 | 816 | 576 | 240 | 24 | 22 | 1 | 1 | BLE | float | UJI | [107] |
| UTS 1 | 9496 | 9108 | 388 | 6.21 | 589 | 1 | 16 | Wi-Fi | int | USYD | [108] |

* Dataset TUJI 1 shall be available online soon.

More information about the datasets may be found in [79] or in their corresponding paper (see references)

detailed analysis of most datasets is performed in [79], including the derivation of the best parameter setting for $k$-NN, which is later used as one of the utilized baselines - baseline $\beta$.

### 3.1.5 Considered Baselines

26 independent datasets are used to fully evaluate the discussed methods in terms of their performance on the aforementioned metrics. Apart from that, they are also compared with up to two $k$-NN baselines in order to show their performance when utilized with the most commonly used matching model in the scope of fingerprinting-based positioning.

### Simple Configuration - $\alpha$

The first baseline, a so-called simple configuration - referred to as baseline $\alpha$ - is obtained as a result of evaluating $k$-NN on each of the previously introduced datasets with the same parameter setting.

The $\alpha$ setting for $k$-NN model is defined with the following parameters. The number of neighbors $k = 1$, the utilized distance metric, used to determine the similarity between two samples, is Sørensen distance [89], [90], while considering positive data representation of each of the RSS measurements. The selection of positive data representation and $k = 1$ was determined by the universal performance and interpretability of the methods, while the Sørensen distance represents a simple, yet efficient metric, which outperforms in terms of positioning the commonly used metrics, such as Euclidean or Manhattan [21]. The numerical representation may be found in Table 3.4, including the building hit $\zeta_{\mathcal{B}}$, floor hit $\zeta_{\mathcal{F}}$, 3D positioning error $\varepsilon_{3D}$, 2D positioning error $\varepsilon_{2D}$ and time required for position prediction $\tau$. The dash (-) is used when not applicable (e.g. for a building hit, where the dataset includes labels of only one building making the building hit always 100%). The $k$-NN configuration $\alpha$ was chosen as a baseline, mainly for its satisfactory general performance without a necessity to sweep over numerous parameters in order to minimize the errors [21], [79].

For reference, the metrics normalized to the baseline $\alpha$, have a subscript $\alpha$ - e.g. $\varepsilon_{3D\alpha}$ represents the 3D positioning error normalized to the baseline $\alpha$.

### Best Configuration - $\beta$

The second baseline, called "Best Coefficient", referred to as baseline $\beta$, includes the results of running $k$-NN on each of the previously introduced positioning datasets, with parameters (data representation, distance metric, and a number of neighbors $k$) that result in the lowest 3D positioning error. The parameters were obtained as a result of a complex sweep described in [79]. The parameter setting and their results are based on [79] (datasets above the middle line), while the results of the datasets that were not evaluated there, were obtained by performing the same sweep (datasets below the middle line), performed by the Author.

The accuracy of the positioning results (2D and 3D positioning error, and floor hit) of baseline $\beta$ are, to the author's knowledge, the best positioning results obtained

**Table 3.4** Simple configuration - Baseline $\alpha$

| Dataset | $\zeta_{\mathcal{B}}$ [%] | $\zeta_{\mathcal{F}}$ [%] | $\varepsilon_{3D}$ [m] | $\varepsilon_{2D}$ [m] | $\tau$ [ms] | Dataset | $\zeta_{\mathcal{B}}$ [%] | $\zeta_{\mathcal{F}}$ [%] | $\varepsilon_{3D}$ [m] | $\varepsilon_{2D}$ [m] | $\tau$ [ms] |
|---|---|---|---|---|---|---|---|---|---|---|---|
| DSI 1 | - | - | 4.45 | 4.45 | 11.4 | TUT 3 | - | 92.26 | 8.73 | 8.45 | 7.6 |
| DSI 2 | - | - | 4.45 | 4.45 | 4.7 | TUT 4 | - | 95.84 | 5.96 | 5.72 | 47.4 |
| LIB 1 | - | 99.62 | 3.22 | 3.20 | 4.6 | TUT 5 | - | 96.84 | 6.26 | 6.13 | 3.9 |
| LIB 2 | - | 99.94 | 2.78 | 2.78 | 4.7 | TUT 6 | - | 99.99 | 1.91 | 1.90 | 30.2 |
| MAN 1 | - | - | 2.83 | 2.83 | 109.3 | TUT 7 | - | 99.31 | 2.24 | 2.06 | 30.0 |
| MAN 2 | - | - | 2.43 | 2.43 | 10.3 | UEX B1 | - | 90.20 | 3.71 | 3.53 | 3.4 |
| MINT 1 | - | - | 2.67 | 2.67 | 38.0 | UEX B2 | - | 94.20 | 4.69 | 4.44 | 4.3 |
| SAH 1 | - | 40.38 | 8.82 | 5.18 | 117.2 | UEX B3 | - | 75.00 | 7.52 | 7.07 | 2.2 |
| SIM 1 | - | - | 3.35 | 3.35 | 81.8 | UJI 1 | 99.91 | 91.99 | 8.25 | 7.34 | 194.8 |
| TIE 1* | - | 66.00 | 5.60 | 3.28 | 111.8 | UJI 2 | 100 | 85.19 | 8.24 | 7.85 | 193.9 |
| TUJI 1 | - | - | 2.97 | 2.97 | 66.7 | UJIB 1 | - | - | 3.03 | 3.03 | 5.6 |
| TUT 1 | - | 93.88 | 7.24 | 6.80 | 12.7 | UJIB 2 | - | - | 4.17 | 4.17 | 4.5 |
| TUT 2 | - | 91.48 | 11.15 | 10.92 | 5.1 | UTS 1 | - | 92.01 | 8.05 | 7.61 | 90.2 |

* The results of TIE 1 are inconsistent with previously published work. This is a result of a labeling error in the original dataset. We resolved this with the authors of the dataset and report the correct value.

on these datasets across the literature.

The numerical representation may be found in Table 3.5, including the building hit $\zeta_{\mathcal{B}}$, floor hit $\zeta_{\mathcal{F}}$, 3D positioning error $\varepsilon_{3D}$, 2D positioning error $\varepsilon_{2D}$ and time required for position prediction $\tau$. The same as for baseline $\alpha$, also here the dash (-) is used when not applicable, and, similarly, the metrics normalized to the baseline $\beta$, have a subscript $\beta$ - e.g. $\varepsilon_{3D\beta}$ represents the 3D positioning error normalized to the baseline $\beta$.

## 3.2 Bit-level Compression Schemes

In this section, the bit-level radio map compression scheme, denoted EWO$K$, is introduced, derived, and thoroughly evaluated. The bit-level compression is realized by reducing the alphabet size of the radio map, similar to SAX and ASAX lossy compression methods introduced in the previous chapter. The alphabet size reduction is realized by performing an element-wise $K$-means to first find the suitable centroid coordinates, which are then considered as the alphabet levels, and consequently clustering the values of the radio map to obtain the reduced radio map representation. The EWO$K$ considers a novel centroid initialization technique, which removes the

**Table 3.5** Best coefficient - Baseline $\beta$

| Dataset | data rep. | distance | $k$ | $\zeta_{\mathcal{B}}$ [%] | $\zeta_{\mathcal{F}}$ [%] | $\varepsilon_{3D}$ [m] | $\varepsilon_{2D}$ [m] | $\tau$ [s] |
|---|---|---|---|---|---|---|---|---|
| **Parameters based on [79]** | | | | | | | | |
| DSI 1 | pow | Sørensen | 11 | - | - | 3.79 | 3.79 | 13.5 |
| DSI 2 | pos | PLGD10 | 9 | - | - | 3.80 | 3.80 | 8.8 |
| LIB 1 | pos | Euclidean$^2$ | 11 | - | 99.94 | 2.48 | 2.48 | 5.4 |
| LIB 2 | pos | PLGD10 | 9 | - | 99.97 | 2.27 | 2.27 | 8.8 |
| MAN 1 | exp | Manhattan | 11 | - | - | 2.06 | 2.06 | 130.2 |
| MAN 2 | exp | Neyman | 11 | - | - | 1.86 | 1.86 | 14.7 |
| SIM 1 | exp | Euclidean$^2$ | 11 | - | - | 2.41 | 2.41 | 91.6 |
| TUT 1 | pos | PLGD40 | 3 | - | 95.51 | 4.45 | 4.23 | 247.0 |
| TUT 2 | pow | Sørensen | 1 | - | 92.05 | 8.10 | 7.80 | 6.0 |
| TUT 3 | pos | Sørensen | 3 | - | 91.42 | 8.55 | 8.17 | 8.7 |
| TUT 4 | pos | PLGD10 | 3 | - | 95.98 | 5.40 | 5.07 | 1086.5 |
| TUT 5 | pos | PLGD40 | 3 | - | 99.59 | 5.26 | 5.25 | 84.9 |
| TUT 6 | pos | Sørensen | 1 | - | 99.99 | 1.91 | 1.90 | 33.8 |
| TUT 7 | pos | Sørensen | 1 | - | 99.31 | 2.24 | 2.07 | 35.4 |
| UJI 1 | pow | Sørensen | 11 | 100 | 95.23 | 6.56 | 6.17 | 218.1 |
| UJI 2 | exp | Neyman | 11 | 100 | 91.37 | 6.09 | 5.60 | 265.6 |
| **Parameters based on sweep** | | | | | | | | |
| MINT 1 | pow | PLGD10 | 11 | - | - | 2.14 | 2.14 | 54.1 |
| SAH 1 | exp | Neyman | 11 | - | 44.23 | 7.20 | 6.03 | 154.6 |
| TIE 1* | pos | LGD | 11 | - | 98.00 | 2.36 | 2.24 | 2442.6 |
| TUJI 1 | exp | Sørensen | 7 | - | - | 2.27 | 2.27 | 68.2 |
| UEX B1 | pos | Euclidean$^2$ | 3 | - | 94.12 | 3.07 | 2.95 | 4.1 |
| UEX B2 | exp | Neyman | 5 | - | 97.10 | 4.16 | 4.01 | 6.4 |
| UEX B3 | pos | Euclidean$^2$ | 3 | - | 65.00 | 6.73 | 6.67 | 3.5 |
| UJIB 1 | exp | Neyman | 11 | - | - | 1.64 | 1.64 | 7.8 |
| UJIB 2 | pos | LGD | 11 | - | - | 2.52 | 2.52 | 7.0 |
| UTS 1 | exp | Neyman | 11 | - | 91.24 | 7.01 | 6.48 | 122.1 |

* The results of TIE 1 are inconsistent with previously published work.
This is a result of a labeling error in the original dataset. We resolved this with the authors of the dataset and report correct value.

effect of randomness from the clustering algorithm and offers robust and reliable performance. Furthermore, an updating mechanism for the centroids is introduced, which enables efficient training database updating with the new samples.

This section first introduces a common, 7-bit benchmark to all available datasets, explains and derives the EWO$K$ radio map reduction scheme, introduces the solutions to the random initialization of $K$-means, derives the dataset-updating mechanism, and finally numerically evaluates the proposed scheme.

## 3.2.1 EWO$K$: Method Derivation and Algorithm

Reducing the size of the individual elements in the radio map is a convenient solution to reducing the radio map size, without affecting its dimensions. The discrete alphabet effectively reduces the resolution of the measurements, yet by doing so, reduces the effects of the uncertainties within the data. In the following paragraphs, the element-wise compression scheme is introduced, along with its core components, mechanics, and capabilities.

### Element-Wise cOmpression using $K$-means (EWO$K$)

EWO$K$ is a ML-based compression scheme, which requires an unsupervised clustering method to find the optimal alphabet levels using all training database features. The author's recent work [14] introduces the concept of element-wise radio map clustering and represents a groundwork for EWO$K$. The clustering is performed by considering every value within the radio map separately, rather than clustering the whole fingerprint. The full positioning scheme with the EWO$K$ compression is visualized in Figure 3.8.

Initially, the training dataset includes the training features in their pre-defined data representation (the radio map), and the corresponding training labels. EWO$K$ requires the following parameters to be defined: the number of clusters $K$, which defines the achieved CR, as is derived further in this section, $K$-means initialization method (also described later), and $K$-means distance metric, using which the centroid levels are optimized. The $K$-means distance is set as Manhattan in this work, as the effect of other distance metrics for $K$-means does not significantly affect the final positioning performance.

The $K$-means clustering [109] is then initialized based on the input parameters,

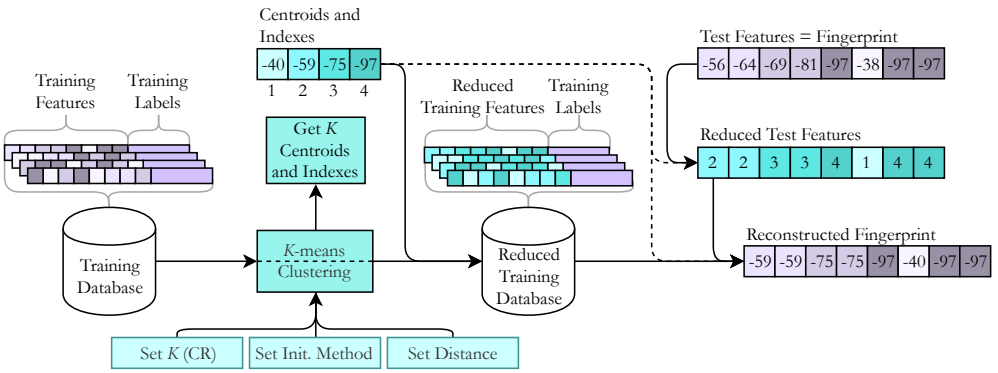


**Figure 3.8** EWO$K$ algorithm

and by processing the whole radio map simultaneously. First, $K$ centroids are selected based on the initialization method, traditionally at the values selected randomly from across all input samples. Iteratively, the algorithm assigns each input sample to its closest centroid based on the minimum value of the chosen distance metric, followed by adjusting the centroid coordinate to minimize the distance to all its assigned samples. $K$-means iteration is interrupted after the centroid coordinates do not change between the iterations (convergence) or after the selected maximum number of iterations is reached. The algorithm then returns the final $K$ centroid values, along with their alphabet indexes.

Using the obtained centroid coordinates, the training database is transformed into the reduced training database, where each element is stored using its corresponding alphabet index. The reduced database can then be stored, reducing the storage requirements. This process can be done offline, not requiring being implemented into the network in real-time. The radio map reconstruction is obtained by simply substituting the alphabet indexes with the corresponding centroid coordinate values.

Next, the online phase, in which user localization in real-time may be performed, follows. The incoming (testing) sample is first transformed to the reduced representation by assigning the closest centroid value to each of its RSS elements and substituting it.

The positioning algorithm, such as $k$-NN, can be then executed. The reconstructed training database serves as training data and the newly measured (or testing) fingerprint represents the sample to be localized. In the scope of this thesis, the online phase refers to the compression of the testing database and its positioning

evaluation.

In this work, the effect of the compression is assessed by first performing the positioning task on the original dataset, followed by the database's compression, and performing the positioning task using the reconstructed radio map. The shift in positioning performance determines the effect of the compression on the quality of the radio map.

The novelty of EWO$K$ is based on implementing the element-wise radio map reduction using $K$-means. Traditionally, $K$-means is utilized in the scope of positioning to cluster the individual fingerprints as whole arrays of RSS measurements, as in [110]–[112], to reduce the number of samples entering the positioning model. Additionally, EWO$K$ scheme incorporates non-random initialization schemes for $K$-means, showing improved reliability of the achieved centroid selection and consequently of the positioning performance itself.

The $K$-means clustering is considered the only clustering mechanism due to its algorithmic simplicity and robustness in the given scenario. Density-based clustering techniques, such as Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [15] are unsuitable for 1-dimensional data, while more complex models, such as the Gaussian Mixture model [113], require more than a single parameter to characterize each cluster (reducing the compression performance), apart from the severely more complex algorithm.

### Compression Ratio of EWO$K$

The Compression Ratio (CR) of EWO$K$ depends on the format used for the storage of the original dataset and the number of $K$ in $K$-means, that is set as the input for EWO$K$. In case the original dataset is stored as a set of integers, each number is most commonly stored as 32 bits (4 bytes) allowing for representation of values between 0 and $4,294,967,295$ ($2^{32}-1$) or alternatively values in the range of $\pm 2^{31}-1$ (resulting into values between $-2,147,483,647$ and $2,147,483,647$). The memory requirements are even higher for data stored as a float.

In the scope of RSS fingerprinting-based positioning, the values stored are of a much smaller range, than the aforementioned storage formats commonly enable storing. Based on Institute of Electrical and Electronics Engineers (IEEE) 802.11 wireless Local Area Network (LAN) standard for radio resource measurements [114], ETSI EN 302 502 [115] and ETSI EN 300 328 [116] specifications, the Wi-Fi an-

tenna transmit power may obtain values of 20 dBm at most for 2.4 GHz bands and 30 dBm at most for 5 GHz bands. The highest values considered for Wi-Fi RSS are approximately 10 dBm. The minimum detected signal strength for Wi-Fi signals is usually measured at around −100 dBm (this value may vary based on the manufactured device).

Based on this and considering, that the most common values stored in the presented fingerprinting datasets are in the range from −30 dBm to −100 dBm, it can be assumed, that the values outside of this range are not necessary in terms of Indoor Positioning System (IPS). Therefore, these values may be sufficiently represented by using only 7 bits, as 7 bits cover 128 values ($2^7$ usually considering to cover a range from 0 to 127).

As covered in 3.1.4, the datasets considered in the scope of this work are stored as integers (denoted as an int in Table 3.3) or floats. Integer format is used for whole number values (belonging to $\mathbb{Z}$) and float is used for the real-valued numbers (belonging to $\mathbb{R}$). Notably, based on the SotA standards, the RSS values are stored as whole numbers. The reason 10 of the mentioned datasets are stored as the float is mainly caused by pre-processing of their data (e.g. averaging, interpolation, etc.), which is further described in [79].

In order to create a common format for all datasets, all datasets' radio maps were stored as whole numbers within the range of 128 consecutive numbers (values expressed as $\mathbb{R}$ were rounded to their closest value in $\mathbb{Z}$), creating a 7-bit benchmark. Although the accuracy of the stored values has changed, the impact on positioning performance is not significantly affected. In order to demonstrate this, the positioning accuracy of all datasets originally stored as floats was evaluated using a simple setting (the same as baseline $\alpha$) of $k$-NN.

Table 3.6 visualizes the 3D positioning error $\varepsilon_{3D}$ and floor hit $\zeta_{\mathcal{F}}$ of the float-based datasets first in the original format, followed by the reduced, 7-bit representation in "Integer Format". Furthermore, the table shows normalized results with "Float Format" as a reference. The aggregated results in the bottom row of the table show that the average floor hit $\tilde{\zeta}_{\mathcal{F}}$ remained unchanged and the average 3D positioning error $\tilde{\varepsilon}_{3D}$ increased by 1% across the considered datasets. As such, the 7-bit benchmark for evaluating the $CR_{bit}$ is considered for all datasets, regardless of their original format.

Given the core of the EWO$K$ is its ability to find a lower number of values needed for the storage of RSS data and consequently lead to the bit-level compression, the

**Table 3.6** Float/integer data representation using baseline $\alpha$ configuration

| | Float Format | | Integer Format | | Float vs Integer | |
|---|---|---|---|---|---|---|
| Dataset | $\zeta_{\mathcal{F}}$ [%] | $\varepsilon_{3D}$ [m] | $\zeta_{\mathcal{F}}$ [%] | $\varepsilon_{3D}$ [m] | $\tilde{\zeta}_{\mathcal{F}}$ [-] | $\tilde{\varepsilon}_{3D}$ [-] |
| MAN 2 | - | 2.43 | - | 2.41 | - | 0.99 |
| MINT 1 | - | 2.67 | - | 2.80 | - | 1.05 |
| TUT 1 | 93.88 | 7.24 | 93.88 | 7.25 | 1.00 | 1.00 |
| TUT 2 | 91.48 | 11.15 | 91.48 | 11.15 | 1.00 | 1.00 |
| TUT 5 | 96.84 | 6.26 | 97.05 | 6.30 | 1.00 | 1.01 |
| UEX B1 | 90.20 | 3.71 | 89.22 | 3.76 | 0.99 | 1.01 |
| UEX B2 | 94.20 | 4.69 | 92.75 | 4.96 | 0.98 | 1.06 |
| UEX B3 | 75.00 | 7.52 | 75.00 | 7.57 | 1.00 | 1.01 |
| UJIB 1 | - | 3.03 | - | 3.03 | - | 1.00 |
| UJIB 2 | - | 4.17 | - | 4.10 | - | 0.98 |
| Average | | | | | 1.00 | 1.01 |

CR from the Equation 3.7 may be specified for EWO$K$ as $CR_{EWOK}$:

$$CR_{EWOK} = \frac{7}{ceil(log_2(K))} \tag{3.11}$$

where the function $ceil(\cdot)$ represents the rounding-up function to the nearest higher whole number and the number $K$ is based on $K$-means input. Consequently, the value of CR may achieve only certain levels, as denoted in Figure 3.9, based on the chosen $K$.

The number of clusters $K$ used in EWO$K$ is the maximum number of possible values that may be stored in the dataset, which is linked to the number of bits (here denoted as $n$) that are required for the storage of these values. In general, this dependency may be expressed as:

$$n = log_2(K) \tag{3.12}$$

For example, if $K = 32$, which is the maximum number of possible values in the dataset, the number of bits $n$ that is required for its storage is 5. On the other hand, also other values exist, that may be only expressed in 5 bits or more. E.g. to express

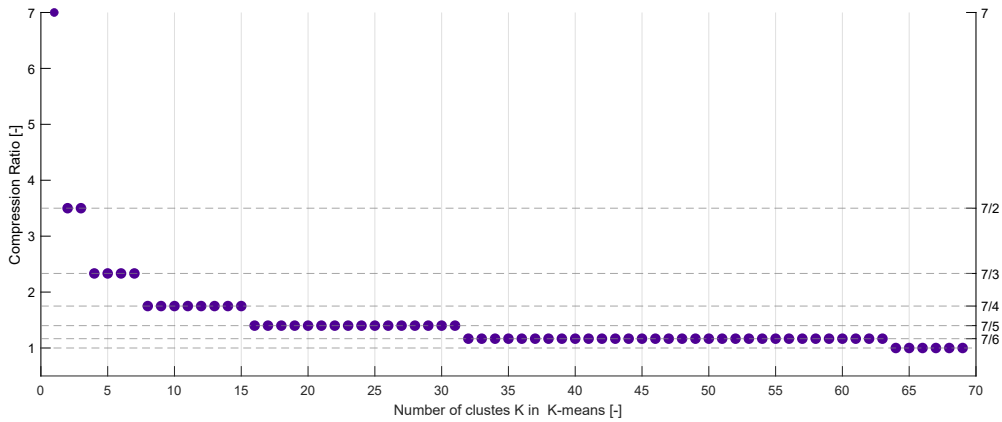


**Figure 3.9** Compression ratio of EWO$K$ dependency on number of clusters $K$

$K = 30$ it is also necessary to use 5 bits. Therefore, the optimal choice for the values of $K$ in $K$-means in terms of EWO$K$, is the value rounded up to the $2^n$ (resulting into $n$ ranging from 1 to 6). Values of $n$ higher than 6 lead to no compression.

The higher the number of $K$ in $K$-means the higher the precision of the stored values and the lower the CR. As discussed later in this thesis, lowering of the precision of the values used for storing RSS data does not always lead to the degradation in positioning accuracy, but it often does.

### Random vs. Non-random initialization

The general implementation of $K$-means clustering is based on the random initialization of the algorithm. This leads to the differentiating results between the runs of $K$-means and consequently of EWO$K$. This is further referred to as a "different run different result" problem. The performance of random initialization of $K$-means is studied e.g. in [117].

The improved initialization algorithm for $K$-means, denoted $k++$ [118], is an augmentation of the "Furthest point heuristic" algorithm [119] by including the randomization. $k++$ randomly selects the first centroid from the population and assigns each subsequent centroid randomly selected from across the remaining samples with the probability proportional to their respective distance to the existing centroids. According to [118], the method increases both convergence speed and accuracy of $k$-means, when compared to the randomly initialized algorithm. Nevertheless, the effect of randomness still occurs, leading to a more stable, yet still uncertain centroid

distribution.

In order to address the issue with the random initialization of $K$-means, 6 non-random initialization methods $\psi$ were proposed for EWO$K$ scheme. Each of them leads to the same result each time the method is run and therefore is not affected by the "different run, different result" problem.

The extent of the random effect and therefore the uncertainty of the final positioning performance is visualized in Figure 3.10. The positioning performance of EWO$K$ with the generic random initialization, and with k++ random initialization is compared with the proposed non-random initiation method $\psi_{xtr}$, introduced later in this section. The comparison is shown over a run that spans the number of clusters from $K = 2$ to $K = 25$.

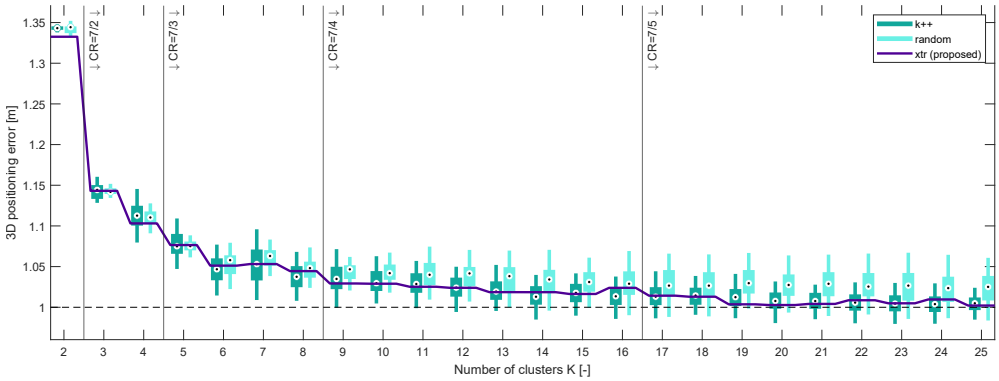


**Figure 3.10** Comparison of achieved $\tilde{\varepsilon}_{3D\alpha}$ for varying $K$, based on $K$-means initialization method

Each boxplot in Figure 3.10 was obtained by repeating the EWO$K$ compression with the given initialization setting for each dataset 25 times, followed by the $k$-NN positioning using the $\alpha$ setting. The obtained $\varepsilon_{3D}$ was then normalized using the corresponding dataset's $\alpha$ baseline. The normalized results were then sorted and averaged across the considered datasets so that the best results across runs are averaged among themselves and so do the worst ones. By doing so, we obtain an expected distribution of errors regardless of the individual scenario. The results of the proposed $\psi_{xtr}$ initialization are normalized towards the benchmark performance and aggregated across the datasets as well. Figure 3.10 further marks the CR thresholds based on the number of clusters $K$.

The proposed initialization methods are denoted as $\psi_{max}$, $\psi_{min}$, $\psi_{xtr}$, $\psi_{imax}$, $\psi_{imin}$ and $\psi_{ixtr}$. The individual initialization methods are based on the distribution (CDF)

of the training samples of original data as pre-defined levels. The recommended choice of the methods shall depend on the given dataset and the performance achieved on the training set. Nevertheless, for a fair evaluation in terms of this thesis, the results are obtained when using the same methods in the comparison for all datasets (no matter their individual performance), if not stated otherwise.

The first of the proposed initialization methods, denoted as $\psi_{max}$, is estimating the $K$ initial values for $K$-means by dividing the $y$-axis of CDF of the given dataset into $K$ segments of the same size, where $K-1$ borders between them define the $K-1$ initial starting levels at their intersections with the CDF (its $x$-axis value), with $K^{th}$ level being assigned to the maximum value, therefore the name of this method.

The second initialization method, denoted as $\psi_{min}$, derives the first $K-1$ levels the same way as $\psi_{max}$, but the last, $K^{th}$ level, is set as the minimum value in the dataset (floor level).

The third initialization method, denoted as $\psi_{xtr}$ ("extreme"), divides the $y$-axis of the CDF into $K-1$ segments of the same size, where $K-2$ borders between them define the first $K-2$ initial starting points, with $K-1^{th}$ level being the maximum probability and $K^{th}$ initial starting point is the minimum one.

In contrast to the usage of equidistant segments for the initialization, the next three initialization methods use incremental division of the distribution with linearly increasing segment sizes. The $\psi_{imax}$ and $\psi_{imin}$ divide the distribution into $N$ intervals of the same size, where the number of intervals corresponds to:

$$N = \sum_{i=0}^{K-1} K - i \tag{3.13}$$

The first $K-1$ starting levels of the CDF are then assigned from the top of the distribution with the first point being the maximum value minus one interval, the second one being spaced 2 intervals below the first one, etc. The initial centroid values are then set as the $x$-values at the intersections of the defined levels and the CDF.

The first starting point for $\psi_{imax}$ and $\psi_{imin}$ are then assigned as the maximum and minimum values within the dataset, respectively.

The last initialization methods denoted as $\psi_{ixtr}$, divides the distribution into $M$ intervals, where $M$ is obtained as:

$$M = \sum_{i=0}^{K-2} K - i \tag{3.14}$$

with the consecutive assignment corresponding to the one of $\psi_{imax}$ and $\psi_{imin}$, with the two last centroid values corresponding to the maximum and minimum values within the dataset.

Figure 3.11 depicts the initial starting levels (as solid lines) and their respective final levels (dashed lines of the same colors) for all 6 proposed initialization methods $\psi$ (when $K = 4$).

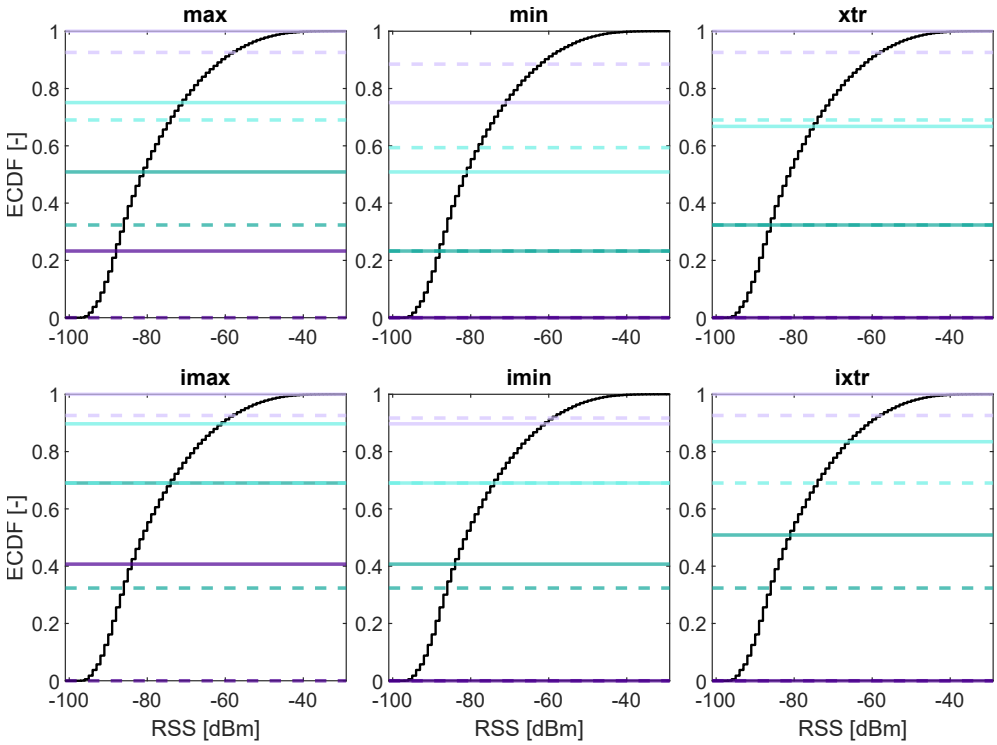


**Figure 3.11** Visualization of initial centroids (solid lines) and their final levels (dashed lines) for the TUJI 1 dataset.

## Dataset-Updating Mechanism

One of the challenges connected with applying a lossy compression mechanism onto the radio map is the inability to incorporate additional samples to the reduced dataset

without introducing a bias. One such example may occur when applying EWO$K$ onto the dataset while determining the optimal centroid values based on the available data, and obtaining an additional batch of training data tardily (e.g. after realizing one part of the deployment was not surveyed properly and repeating the measurement campaign). Removing the faulty samples from the dataset is straightforward while adding the new ones using the previously determined centroids will not reflect the new data distribution, resulting in possible sub-optimal centroid values.

In the following paragraphs, a centroid updating scheme for element-wise compression scheme, such as EWO$K$, is introduced, enabling unbiased updating of the reduced radio map with new samples. The centroid updating mechanism was first introduced in the author's publication [14].

The mechanism of Adaptive $K$-means is initialized after the clustering is performed and $K$ centroids are found (see Figure 3.8). During the training dataset transformation to its reduced representation, the dataset-updating mechanism counts the number of occurrences of each alphabet index in the compressed radio map. The algorithm then performs the following steps every time the new sample is designated to be added to the existing radio map:

1. Find the closest centroid values to each element of the sample's feature vector.
2. Update the existing centroid coordinates and measurement counts for every RSS element in the new sample $s$ as:

$$C_n = \frac{C_n \cdot N_n + f_{s,a}}{N_n + 1} \tag{3.15}$$

$$N_n = N_n + 1 \tag{3.16}$$

   where $f_{s,a}$ denotes the measured RSS value of the new sample $s$ at $a^{th}$ AP, $C_n$ denotes the closest centroid value assigned to that element, while $N_n$ denotes the current number of occurrences of $n^{th}$ alphabet symbol within the reduced radio map.
3. Add the newly compressed sample to the reduced training database.

The algorithm is visualized in Figure 3.12, where the "Update Centroid Coordinates" block refers to Equations 3.15 and 3.16.

The immediate effect of the algorithm is that the centroid coordinates are slightly shifted every time a new sample is added to the database (weighted by the count

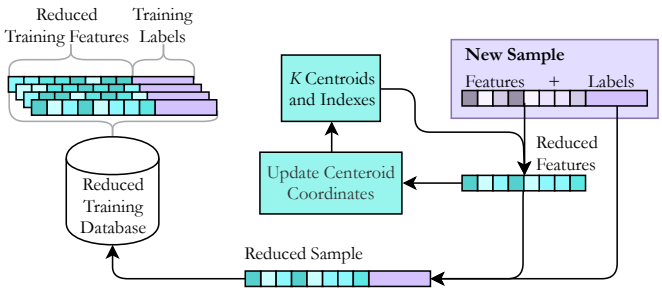


**Figure 3.12** Adaptive $K$-means mechanism

of the existing occurrences of that symbol in the reduced database) so that they represent the overall population of the reduced radio map. The main advantage of the algorithm is that the centroids' values represent all samples within the database without bias. The downsides of the Adaptive $K$-means include additional processing requirements, slight storage overhead, and an inability of the algorithm to account for reversing the effects of shifting the centroid values (e.g. when removing the faulty fingerprints from the database).

## 3.2.2 Numerical Evaluation

In this section, the numerical results of the proposed bit-level compression scheme denoted as EWO$K$ are presented. For the fairness of comparison, all obtained performance metrics are obtained using the same settings as either $\alpha$ or $\beta$ baselines and normalized to their respective baseline.

The first part of the evaluation includes the visualization of the performance shift as the result of applying EWO$K$ compression scheme with the proposed initialization methods. For each initialization, the number of clusters $K$ equal to 8, 16, and 32, achieving the $CR_{EWOK}$ of 7/3, 7/4, and 7/5, respectively, was considered. The $k$-NN setting and the normalization baseline $\alpha$ are considered. Figure 3.13 compactly lists the normalized $3D$ positioning error $\tilde{\varepsilon}_{3D\alpha}$ across all 26 considered datasets. The individual results show that certain datasets suffered a slight performance decrease (characterized by the purple color), while others experienced an improvement in positioning performance (green color). In many cases, the compression did not affect the positioning performance (white). The bottom row in Figure 3.13 (AVG.) shows the averaged aggregated results across all datasets for each initialization method

and each considered $K$, showing light-purple coloration of the points, denoting a slight increase in $3D$ positioning error $\varepsilon_{3D}$ on average. The trade-off, considering the achieved compression, is favorable, especially considering the unaffected performance when applying $\psi_{xtr}$ initialization with $K = 32$.

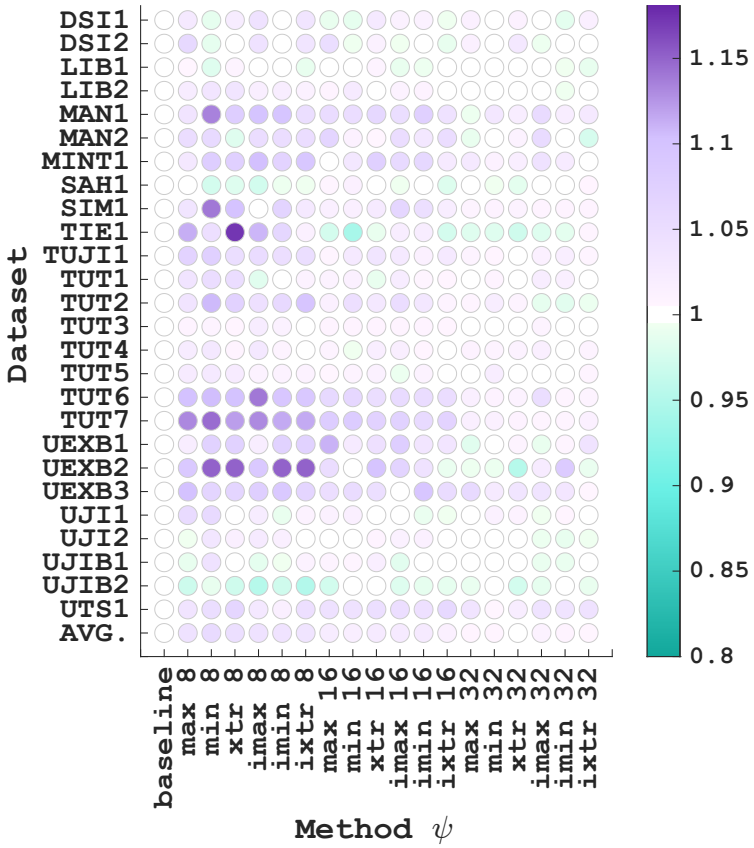


**Figure 3.13** Sweep over initialization methods - 3D positioning error

The numerical results towards the same baseline, namely $\alpha$, are provided in Table 3.7, where the normalized floor hit $\tilde{\zeta}_{\mathcal{F}\alpha}$, normalized $3D$ positioning error $\tilde{\varepsilon}_{3D\alpha}$, and normalized $2D$ positioning error $\tilde{\varepsilon}_{2D\alpha}$ are evaluated for EWO$K$ with $K = 8$, denoted EWO8, $K = 16$, denoted EWO16, and $K = 32$, denoted EWO32, using $\psi_{xtr}$ initialization method. Note that Table 3.7 does not include the evaluation of $\tilde{\zeta}_{\mathcal{B}\alpha}$, because the building hit $\zeta_{\mathcal{B}}$ was not affected by applying the EWO$K$ compression. The numerical results show an average increase in $\varepsilon_{3D}$ by 5% and a decrease in floor hit $\zeta_{\mathcal{F}}$ by 2% when considering EWO8, yet when increasing the number of clusters $K$ to 32 the performance deterioration disappears.

**Table 3.7** Results of EWOK - Simple baseline

| | EWO8, CR=7/3 | | | EWO16, CR=7/4 | | | EWO32, CR=7/5 | | |
|---|---|---|---|---|---|---|---|---|---|
| Dataset | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] |
| DSI 1 | - | 1.02 | 1.02 | - | 1.03 | 1.03 | - | 1.01 | 1.01 |
| DSI 2 | - | 1.00 | 1.00 | - | 1.01 | 1.01 | - | 1.03 | 1.03 |
| LIB 1 | 1.00 | 1.01 | 1.01 | 1.00 | 1.01 | 1.01 | 1.00 | 1.00 | 1.00 |
| LIB 2 | 1.00 | 1.04 | 1.04 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |
| MAN 1 | - | 1.08 | 1.08 | - | 1.06 | 1.06 | - | 1.02 | 1.02 |
| MAN 2 | - | 0.98 | 0.98 | - | 1.00 | 1.00 | - | 1.01 | 1.01 |
| MINT 1 | - | 1.08 | 1.08 | - | 1.07 | 1.07 | - | 1.02 | 1.02 |
| SAH 1 | 0.97 | 0.98 | 0.82 | 1.00 | 1.00 | 1.01 | 1.00 | 0.99 | 1.00 |
| SIM 1 | - | 1.10 | 1.10 | - | 1.03 | 1.03 | - | 1.01 | 1.01 |
| TIE 1 | 0.97 | 1.17 | 1.17 | 1.06 | 0.99 | 1.11 | 1.00 | 0.97 | 1.00 |
| TUJI 1 | - | 1.05 | 1.05 | - | 1.04 | 1.04 | - | 1.00 | 1.00 |
| TUT 1 | 0.99 | 1.05 | 1.07 | 1.00 | 0.99 | 0.99 | 0.99 | 1.00 | 1.00 |
| TUT 2 | 1.00 | 1.07 | 1.08 | 0.99 | 1.03 | 1.03 | 1.00 | 1.01 | 1.01 |
| TUT 3 | 1.00 | 1.01 | 1.00 | 1.00 | 1.01 | 1.00 | 1.00 | 1.00 | 1.00 |
| TUT 4 | 1.00 | 1.01 | 1.00 | 0.99 | 1.02 | 1.00 | 1.00 | 1.01 | 1.01 |
| TUT 5 | 1.00 | 1.02 | 1.02 | 1.00 | 1.01 | 1.02 | 1.00 | 1.00 | 1.00 |
| TUT 6 | 1.00 | 1.10 | 1.11 | 1.00 | 1.05 | 1.05 | 1.00 | 1.01 | 1.01 |
| TUT 7 | 1.00 | 1.12 | 1.09 | 1.00 | 1.07 | 1.04 | 1.00 | 1.01 | 1.01 |
| UEX B1 | 0.93 | 1.07 | 1.01 | 0.99 | 1.04 | 1.04 | 1.00 | 1.01 | 1.00 |
| UEX B2 | 0.95 | 1.15 | 1.15 | 1.02 | 1.10 | 1.12 | 1.02 | 0.95 | 0.96 |
| UEX B3 | 0.91 | 1.07 | 1.04 | 0.98 | 1.05 | 1.06 | 1.00 | 1.04 | 1.05 |
| UJI 1 | 1.00 | 1.00 | 1.01 | 1.00 | 1.00 | 1.01 | 1.00 | 1.01 | 1.01 |
| UJI 2 | 0.99 | 1.02 | 1.01 | 1.00 | 1.01 | 1.01 | 1.00 | 1.00 | 1.00 |
| UJIB 1 | - | 1.00 | 1.00 | - | 1.02 | 1.02 | - | 1.00 | 1.00 |
| UJIB 2 | - | 0.97 | 0.97 | - | 1.00 | 1.00 | - | 0.97 | 0.97 |
| UTS 1 | 0.99 | 1.06 | 1.06 | 1.00 | 1.05 | 1.04 | 1.00 | 1.02 | 1.02 |
| Average | 0.98 | 1.05 | 1.04 | 1.00 | 1.03 | 1.03 | 1.00 | 1.00 | 1.01 |

Similarly, Table 3.8 presents the results when the settings of $k$-NN, as well as the normalization, consider the baseline $\beta$. The EWO$K$ settings are consistent with the previous evaluation. The results do not include the evaluation of $\tilde{\zeta}_{\mathcal{B}\beta}$, because the building hit was not affected. The table shows consistent results with Table 3.7 in terms of the performance shift in terms of the evaluated metrics.

**Table 3.8** Results of EWOK - Best coefficient baseline

| | EWO8, CR=7/3 | | | EWO16, CR=7/4 | | | EWO32, CR=7/5 | | |
|---|---|---|---|---|---|---|---|---|---|
| Dataset | $\tilde{\zeta}_{\mathcal{F}\beta}$ [-] | $\tilde{\varepsilon}_{3D\beta}$ [-] | $\tilde{\varepsilon}_{2D\beta}$ [-] | $\tilde{\zeta}_{\mathcal{F}\beta}$ [-] | $\tilde{\varepsilon}_{3D\beta}$ [-] | $\tilde{\varepsilon}_{2D\beta}$ [-] | $\tilde{\zeta}_{\mathcal{F}\beta}$ [-] | $\tilde{\varepsilon}_{3D\beta}$ [-] | $\tilde{\varepsilon}_{2D\beta}$ [-] |
| DSI 1 | - | 0.96 | 0.96 | - | 0.99 | 0.99 | - | 1.00 | 1.00 |
| DSI 2 | - | 1.01 | 1.01 | - | 1.01 | 1.01 | - | 1.01 | 1.01 |
| LIB 1 | 1.00 | 0.98 | 0.98 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |
| LIB 2 | 1.00 | 1.04 | 1.04 | 1.00 | 1.02 | 1.01 | 1.00 | 1.00 | 1.00 |
| MAN 1 | - | 1.09 | 1.09 | - | 1.06 | 1.06 | - | 1.02 | 1.02 |
| MAN 2 | - | 1.13 | 1.13 | - | 1.02 | 1.02 | - | 1.03 | 1.03 |
| MINT 1 | - | 1.07 | 1.07 | - | 1.01 | 1.01 | - | 0.99 | 0.99 |
| SAH 1 | 1.00 | 1.00 | 0.91 | 1.00 | 1.02 | 1.04 | 1.00 | 1.00 | 0.98 |
| SIM 1 | - | 1.13 | 1.13 | - | 1.06 | 1.06 | - | 1.01 | 1.01 |
| TIE 1 | 0.98 | 1.11 | 1.00 | 1.00 | 1.05 | 1.06 | 1.00 | 1.04 | 1.04 |
| TUJI 1 | - | 1.10 | 1.10 | - | 1.05 | 1.05 | - | 1.02 | 1.02 |
| TUT 1 | 1.00 | 1.02 | 1.04 | 1.00 | 1.03 | 1.04 | 1.00 | 0.99 | 1.00 |
| TUT 2 | 0.99 | 1.18 | 1.16 | 1.01 | 1.12 | 1.10 | 1.00 | 1.03 | 1.01 |
| TUT 3 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 0.99 | 1.00 | 1.00 | 0.99 |
| TUT 4 | 0.99 | 1.04 | 1.01 | 1.00 | 1.02 | 1.01 | 1.00 | 1.01 | 1.01 |
| TUT 5 | 1.00 | 1.04 | 1.04 | 1.00 | 1.03 | 1.02 | 1.00 | 1.01 | 1.01 |
| TUT 6 | 1.00 | 1.10 | 1.11 | 1.00 | 1.05 | 1.05 | 1.00 | 1.01 | 1.01 |
| TUT 7 | 1.00 | 1.12 | 1.09 | 1.00 | 1.07 | 1.04 | 1.00 | 1.01 | 1.01 |
| UEX B1 | 0.93 | 1.24 | 1.18 | 0.95 | 1.14 | 1.04 | 0.99 | 1.03 | 1.03 |
| UEX B2 | 0.99 | 1.11 | 1.10 | 1.00 | 1.03 | 1.01 | 1.01 | 1.00 | 1.00 |
| UEX B3 | 0.97 | 0.99 | 1.02 | 1.08 | 1.03 | 1.07 | 0.92 | 1.08 | 1.15 |
| UJI 1 | 1.00 | 1.06 | 1.05 | 1.00 | 1.02 | 1.02 | 1.00 | 1.02 | 1.03 |
| UJI 2 | 0.98 | 1.05 | 1.07 | 0.99 | 1.01 | 1.02 | 0.99 | 1.01 | 1.01 |
| UJIB 1 | - | 1.06 | 1.06 | - | 1.02 | 1.02 | - | 1.00 | 1.00 |
| UJIB 2 | - | 1.00 | 1.00 | - | 0.98 | 0.98 | - | 1.01 | 1.01 |
| UTS 1 | 1.01 | 1.01 | 1.01 | 1.01 | 1.02 | 1.03 | 0.99 | 1.02 | 1.02 |
| Average | 0.99 | 1.06 | 1.05 | 1.00 | 1.03 | 1.03 | 1.00 | 1.01 | 1.01 |

The numerical evaluation of the proposed bit-level compression scheme denoted Element-Wise cOmpression using $K$-means (EWO$K$) shows that considerable radio map compression can be achieved with minimal to no effect on the positioning performance. The evaluation on 26 independent datasets and 2 different baselines, one of which achieves optimized positioning performance, shows that EWO$K$'s performance is universally stable. The tunable trade-off between the CR and the performance deterioration enables freedom in fine-tuning the positioning system. Moreover, the proposed non-random initialization schemes remove the effect of randomness from the $K$-means clustering, which introduced severe uncertainties to the overall positioning system.

## 3.3 Feature-wise Compression Schemes

In this section, the radio map is compressed in terms of the number of features per fingerprint. The width of the original radio map is determined by the number of APs within the given scenario (see Table 3.3) yet in many cases, the majority of the measurements correspond to the unmeasured RSS values. For example, the radio map of the TUJI 1 dataset consists of 2, 093, 120 individual RSS elements, from which only 149, 825 (7.16%) include actual measurements. Therefore, the radio map includes a lot of redundancies.

The goal of the feature-wise compression schemes is to remove such redundancies. One approach is to filter out the insignificant APs and leave only the "meaningful" ones using either a heuristic (e.g. missing values ratio or low variance filter [120]) or data-driven algorithms, such as data pruning using Decision Tree Ensembles [121], or Bhattacharyya distance [122]. The other approach is to implement a meaningful transformation, which will reduce the width of the radio map matrix.

In this section, two transformation-based algorithms performing feature-wise compression are evaluated, namely PCA [123] and AE [124]. Apart from evaluating the achievable radio map compression and its impact on positioning performance, the goal of this evaluation is to estimate whether breaking the geometry of the deployment using the transformation affects the positioning performance, and to what extent.

### 3.3.1 Methods and Algorithms

In the paragraphs below, the system models implementing the feature-wise radio map compression for PCA and AE are presented. PCA is considered as the solution determining the number of reduced radio map's features, which is then matched by the AE implementation to compare the performance of both methods at the same CR levels. Both PCA and AE consider only the samples within the radio map, thus not being able to find correlations based on sample similarity in real space (i.e. physical coordinates).

#### Principal Component Analysis (PCA)

PCA extracts the principal components from the matrix and uses them to transfer the input matrix into its orthogonal basis [123], [125]. They are obtained via Eigen decomposition of the covariance matrix, or through Singular Value Decomposition (SVD) algorithm, which is computationally more efficient. PCA found its application in data compression, dimensionality reduction [126], and feature extraction [127], either as a stand-alone algorithm or combined with other methods such as in [128], where clustering and PCA were combined to improve positioning accuracy. The compression using PCA is achieved by disregarding the insignificant principal components. Notably, PCA compression mechanism is a non-linear transformation due to the removal of the insignificant components, while the PCA operation itself is linear.

The algorithmic model for feature-wise compression is visualized in Figure 3.14. First, the feature-wise mean $\mu$ is removed from the training features, followed by performing the PCA decomposition on the training radio map (training features) to obtain the principal component coefficients (Coefficients), each sample's representation in the principal component space (Scores), as well as the significance of every principal component characterized by variance explained (Var. Explained). By considering the pre-defined Threshold $\mathcal{T}$, the number of preserved components $n$ (and thus the achieved CR) is determined by iteratively adding the individual variances (sorted from the largest) until the cumulative variance explained is larger than Threshold $\mathcal{T}$. The higher the $\mathcal{T}$, the lower the CR, but the larger the number of considered principal components.

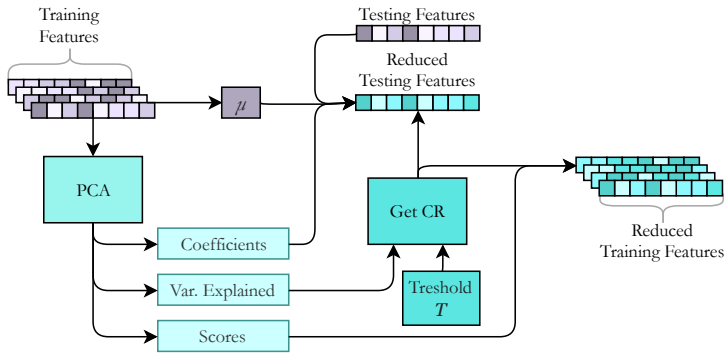


**Figure 3.14** PCA compression algorithm

The reduced training radio map (Reduced Training Features) is then obtained by considering the first $n$ features from the Scores matrix. The Reduced Testing Features are obtained by first removing the feature-wise mean $\mu$ from the Testing Features, followed by their transformation using the reduced matrix of Coefficients (containing only the coefficients of the preserved principal components).

Consequently, PCA is utilized as the dimensionality reduction technique with tunable CR based on the pre-defined Threshold $\mathcal{T}$ that determines the number of preserved coefficients. The detailed description of PCA is defined in the related literature [123], [125], and thus, the mathematical formulation of the decomposition is not repeated here.

### Autoencoder (AE)

The class of the ML methods called AE, first proposed in [124], belongs to the group of unsupervised learning methods. AEs are the NN-based structures, which find useful features within the data and reconstruct the original data from the extracted features, at the same time. In other words, AE is a feed-forward NN approach to reconstruct an output from an input, as they learn to copy their input to their output. There are two parts of every AE, an encoder, which transforms the inputs, and a decoder, which attempts to reconstruct them back [129], [130]. The authors of [129] proposed a semi-supervised scheme for training a variational AE for indoor localization, while [130] developed a deep AE to extract features from the fingerprints.

The objective of the AE training is to perfectly match the input data to the output,

while the differences are used to adjust the inner weights in the model to reduce the error. There is no other requirement on the intermediate layer output. The utilization of classical AE, depicted in Fig 3.15 on the left, is data compression, while the operation on the data represents feature space transformation similar to e.g. PCA, and the new, transformed features represent the coordinates in the new feature space.

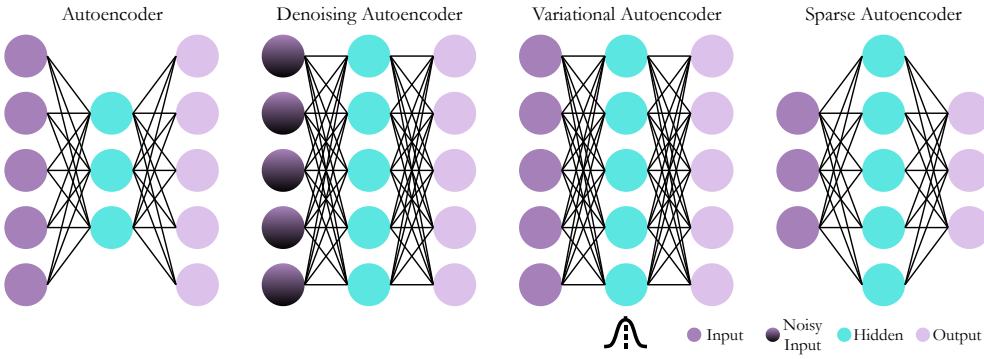


**Figure 3.15** Different architectures of autoencoders

There are numerous variations of the AE, each adjusting its behavior in a specific way to adapt to the desired application.

Denoising AE (Fig 3.15) is designed to artificially remove the noise from the input data [131]. The training is realized by feeding the noisy images, measurements, or any other data to the input of the denoising AE and comparing the output to their noiseless version. The denoising AE can be utilized as a low-complexity filter or stacked with other NN structures.

Variational AE (see Fig 3.15) describes the structures with an additional requirement on the output layer. The objective of the optimization is no longer only to perfectly match the input to the output but to force the intermediate layer variables to correspond to the chosen distribution, most commonly Gaussian. Variational AEs may therefore serve as outlier detectors in addition to the feature extractor properties. In principle, as the model was trained on non-outlier features, all samples corresponding to the same distribution as the training data will be encoded into the chosen distribution representation. The samples from different distributions (e.g. outliers) will be encoded into features, and their distribution will not correspond to the chosen one.

Sparse AEs [132] are a special kind of AE, that have an additional constraint to ensure sparsity of the result to restrict AE from directly applying the identity function between the input and the output. Due to these constraints, sparse AEs are able to consist of more hidden units in the hidden layer than in the input (as well as output) layer (see Fig 3.15, right), being able to extract a larger number of features from the data. The sparsity can be achieved by forcing all weights of the hidden layer to a pre-defined constant as in [133] or by applying $L1$ penalty as in [134] to the hidden layer.

Due to the rapid development of computational methods in NN during the last decades, the deep AEs were enabled. The addition of a larger number of intermediate layers enables deeper feature extraction and more efficient compression than the 3-layer AEs are capable of, with the significant complexity trade-off. Deep AEs can be built and trained in a straightforward manner or be created by stacking basic AE structures. Modern AEs freely incorporate convolutional architectures [135], [136], or temporal layers such as Long-Short Term Memory (LSTM) [135] as well.

The AE implemented in this section is a classical AE with a single hidden layer. Classical AE is considered as the only task of the AE is to compress and reconstruct the data, without additional regularizations, or losses (such as in variational AE). Deep AEs are not considered due to their increased complexity, and since the aim is to compare the AE performance to the PCA with similar complexity (as classical AE can be initialized with the weights obtained as the Coefficients of PCA). Its architecture is visualized in Figure 3.16 and is fully determined by the number of APs $|AP|$ in the original dataset and the number of preserved coefficients $n$, determined by the PCA algorithm at the pre-defined Threshold $\mathcal{T}$.

The encoder structure considered in this work contains $|AP|$ inputs and $n$ neurons in the hidden dimensions, each with Rectified Linear Unit (ReLU) activation function. Its output represents the encoded data, and by running inference on the original training features, the encoder output represents the reduced training feature representation. Similarly, when running inference on the original testing features, the encoder outputs the reduced testing features.

Since AE is a NN, it requires training to adjust to the data. For this purpose, the decoder is implemented, which considers the encoder outputs as its inputs and consists of $|AP|$ neurons with linear activation. The AE training is realized by considering the training features as both features and labels, forcing the NN to first

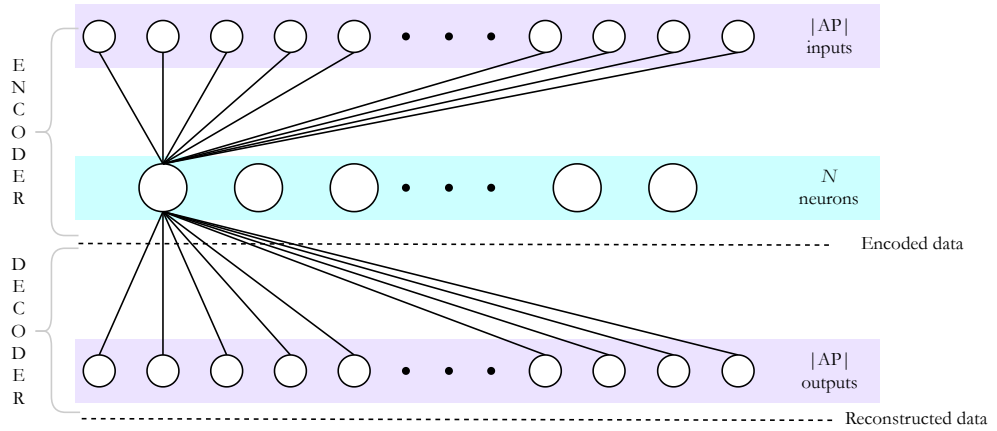


**Figure 3.16** Utilized autoencoder architecture

decompose each training sample's features to a reduced representation using the encoder, and then reconstruct the encoded data back to the training sample's features. After training, the decoder is not utilized anymore.

The AE hyperparameters, apart from its dimensions and activations, include MAE loss function, Adam optimizer with a learning rate of 0.001, and an Early Stopping mechanism with the patience of 3, which interrupts the training when the validation loss stops decreasing. The validation data was selected as 20% of the original training samples (see Table 3.3), which are otherwise not considered in the training procedure. The maximum number of epochs is set to 300, although all datasets converged significantly sooner.

### 3.3.2 Numerical Evaluation

In this section, a numerical evaluation of the two implemented feature-wise compression schemes is provided. To provide a fair evaluation in the scope of this thesis, all results are normalized and compared to the benchmark $\alpha$, the $k$-NN setting used for the positioning on the feature-wise compressed data is also the same as for the $\alpha$ benchmark.

As the result of PCA and AE transforming the radio map's feature to a new representation, the existence of negative coefficients becomes possible, as opposed to the original database, where the data representation transforms the negative values in dBm to strictly non-negative values (positive data representation). Consequently,

the considered Sørensen distance metric needs to be adjusted using absolute values to correctly represent the individual elements' contributions as:

$$dist_{Sorensen,abs}(f_i, \overline{f}_i) = \frac{\sum_{a=1}^{|AP|} |f_{i,a} - \overline{f}_{i,a}|}{\sum_{a=1}^{|AP|} |f_{i,a}| + |\overline{f}_{i,a}|} \tag{3.17}$$

where $f_\mathbf{i}$ and $\overline{f}_\mathbf{i}$ denote the $i^{th}$ sample's RSS feature vector and its estimate, respectively, while $f_{i,a}$ and $\overline{f}_{i,a}$ denote their $a^{th}$ element's numerical value, $|\cdot|$ denotes absolute value.

The evaluation of the achieved CR and the positioning performance trade-off is summarized in Figure 3.17, visualizing the behavior of the aggregated normalized 3D positioning error $\tilde{\varepsilon}_{3D\alpha}$ and the CR when varying the Threshold $\mathcal{T}$. The results show, that PCA decomposition provides more robust features for the $k$-NN than the utilized AE with approximately 10% lower positioning error across the considered CRs. This figure also shows that the feature-wise compression schemes result in increased positioning errors when compared to the baseline results or bit-level compression (such as EWO$K$ introduced in Section 3.2) while achieving substantially higher CRs.

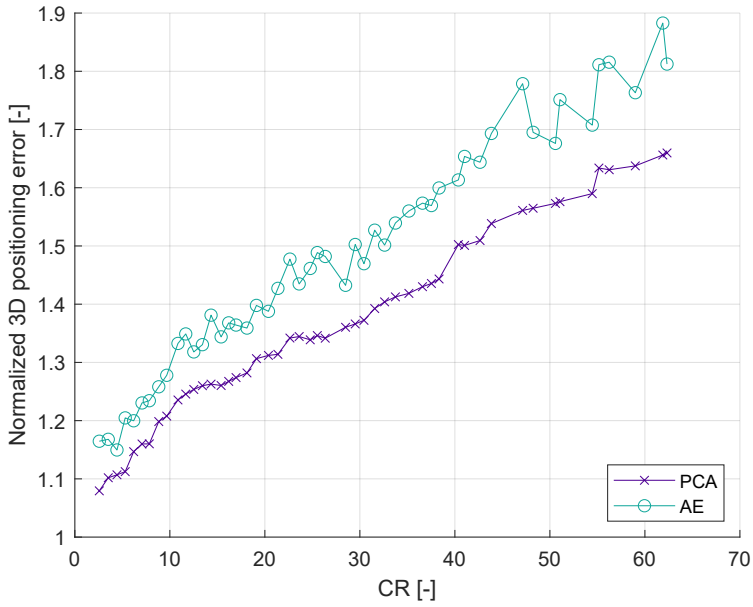


**Figure 3.17** Positioning error and CR trade-off of PCA and AE

Further evaluation provides insights into the impact of the PCA and AE compres-

sion on the aggregated normalized time required for positioning $\tilde{\tau}_\alpha$ while utilizing $k$-NN. Figure 3.18 shows that reducing the number of considered features strongly boosts the speed of the $k$-NN algorithm in both cases, as calculating the distances between the samples is the most demanding task of the algorithm. The normalized prediction time of 0.01 denotes 100-fold reduction of the positioning time when compared to the baseline $\alpha$. The prediction time decreases exponentially with increasing CR as less AP-wise features have to be compared. Figure 3.18 also shows that the normalized prediction time $\tilde{\tau}_\alpha$ starts saturating, denoting the minimum time required to obtain the label estimates. The results further indicate that the estimation time of PCA compression is slightly longer than that of the AE compression, while still drastically reducing the positioning time.

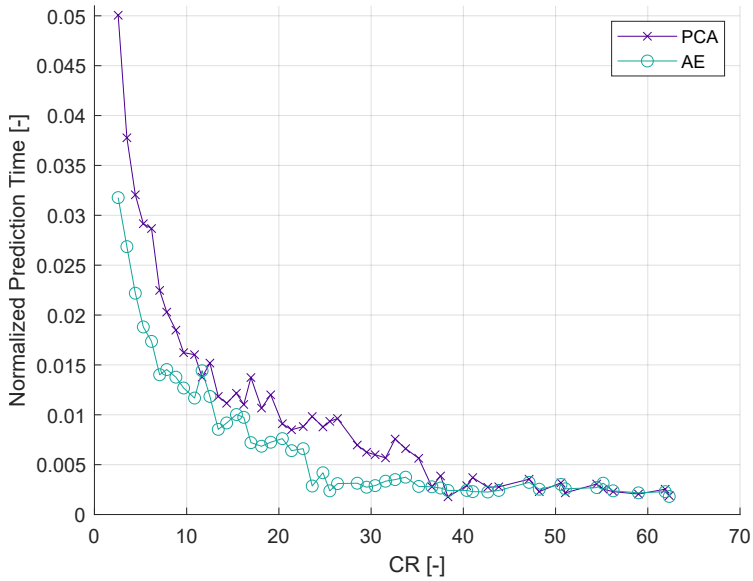


**Figure 3.18** Positioning time and CR trade-off of PCA and AE

Table 3.9 provides the numerical results across all considered datasets for both AE and PCA at the Threshold $\mathcal{T} = 90$, showing the common CR, and the normalized floor hit $\tilde{\zeta}_{\mathcal{F}\alpha}$, 3D positioning error $\tilde{\varepsilon}_{3D\alpha}$, 2D positioning error $\tilde{\varepsilon}_{2D\alpha}$ and positioning time $\tilde{\tau}_\alpha$ of each model, including the aggregated results in the bottom row of the table.

The individual results show that, in the majority of cases, the positioning performance was deteriorated by applying the PCA compression (by 24% on average), while for several datasets (e.g. TUT 4 or UTS 1) the error increased only slightly,

**Table 3.9** Numerical results for PCA and AE at $\mathcal{T} = 90$

| | | PCA | | | | AE | | | |
|---|---|---|---|---|---|---|---|---|---|
| Dataset | CR | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] |
| DSI 1 | 9.2 | - | 1.18 | 1.18 | 0.08 | - | 1.25 | 1.25 | 0.08 |
| DSI 2 | 9.2 | - | 1.19 | 1.19 | 0.28 | - | 1.27 | 1.27 | 0.28 |
| LIB 1 | 43.5 | 1.00 | 0.98 | 0.98 | 0.02 | 0.98 | 1.15 | 1.13 | 0.04 |
| LIB 2 | 32.8 | 0.99 | 1.47 | 1.45 | 0.12 | 0.99 | 1.69 | 1.67 | 0.11 |
| MAN 1 | 5.6 | - | 1.12 | 1.12 | 0.04 | - | 1.55 | 1.55 | 0.03 |
| MAN 2 | 14 | - | 1.33 | 1.33 | 0.04 | - | 1.75 | 1.75 | 0.07 |
| MINT 1 | 5.5 | - | 1.07 | 1.07 | 0.03 | - | 1.22 | 1.22 | 0.03 |
| SAH 1 | 38.75 | 1.33 | 1.30 | 2.47 | 0.09 | 1.29 | 1.04 | 1.79 | 0.06 |
| SIM 1 | 2.7 | - | 1.04 | 1.04 | 0.03 | - | 1.32 | 1.32 | 0.04 |
| TIE 1 | 34.1 | 0.06 | 1.50 | 2.32 | 0.09 | 0.30 | 1.71 | 1.21 | 0.05 |
| TUJI 1 | 15.5 | - | 1.16 | 1.16 | 0.18 | - | 1.27 | 1.27 | 0.06 |
| TUT 1 | 20.6 | 0.97 | 1.17 | 1.15 | 0.08 | 0.94 | 1.24 | 1.17 | 0.06 |
| TUT 2 | 27.2 | 0.91 | 1.10 | 1.03 | 0.21 | 0.89 | 1.21 | 1.22 | 0.14 |
| TUT 3 | 32 | 1.00 | 1.09 | 1.02 | 0.12 | 0.98 | 1.10 | 1.03 | 0.11 |
| TUT 4 | 30.1 | 1.00 | 1.03 | 1.02 | 0.18 | 0.99 | 1.05 | 1.04 | 0.13 |
| TUT 5 | 34.9 | 0.99 | 1.14 | 1.16 | 0.10 | 0.98 | 1.20 | 1.22 | 0.15 |
| TUT 6 | 34.3 | 1.00 | 1.44 | 1.45 | 0.06 | 1.00 | 1.39 | 1.39 | 0.04 |
| TUT 7 | 36.4 | 0.99 | 1.46 | 1.37 | 0.04 | 0.99 | 1.44 | 1.34 | 0.04 |
| UEX B1 | 30 | 0.68 | 2.18 | 1.93 | 0.00 | 0.54 | 2.21 | 2.03 | 0.00 |
| UEX B2 | 15 | 0.77 | 1.83 | 1.61 | 0.10 | 0.79 | 2.02 | 1.85 | 0.00 |
| UEX B3 | 6 | 0.98 | 1.13 | 1.19 | 0.00 | 0.91 | 1.16 | 1.19 | 0.00 |
| UJI 1 | 15.8 | 0.98 | 1.05 | 1.06 | 0.14 | 0.98 | 1.10 | 1.10 | 0.08 |
| UJI 2 | 15.3 | 1.00 | 1.02 | 1.07 | 0.17 | 1.02 | 1.04 | 1.08 | 0.10 |
| UJIB 1 | 2.2 | - | 1.05 | 1.05 | 0.11 | - | 1.09 | 1.09 | 0.21 |
| UJIB 2 | 1.8 | - | 1.07 | 1.07 | 0.09 | - | 1.11 | 1.11 | 0.01 |
| UTS 1 | 17.3 | 1.04 | 1.04 | 1.04 | 0.20 | 1.02 | 1.08 | 1.07 | 0.14 |
| Average | 20.38 | 0.92 | 1.24 | 1.29 | 0.10 | 0.92 | 1.33 | 1.32 | 0.08 |

despite reducing the radio map size 30.1 and 17.3 times, respectively. The AE was capable of outperforming the PCA solution in individual cases ($\tilde{\varepsilon}_{3D\alpha}$ of SAH 1), while globally providing poorer results.

Nevertheless, the utilized AE solution consists of a single architecture, that performs the compression within a single densely connected layer, as well as the reconstruction. Furthermore, the majority of the fingerprinting datasets consist of few thousand or fewer training samples, limiting the capabilities of the NN to properly adapt to the specific scenario. Furthermore, data-driven solutions are, as the name suggests, data-dependent, and fine-tuning the AE in terms of the NN architecture, regularization, optimizer, etc. to each dataset separately would undoubtedly lead to further increase of the positioning performance while fine-tuning the NNs is not within the scope of this thesis.

## 3.4 Sample-wise Compression Schemes

The third dimension, in which the radio map can be reduced, is in the number of samples within the dataset. Despite the fact that the rule of thumb in ML says that the more the data, the more robust the training, it is not always the case when considering the indoor positioning scenario or $k$-NN. The indoor positioning datasets often contain outliers or faulty measurements, which can negatively affect the model training, especially if the model is a simple matching algorithm, such as $k$-NN. $k$-NN does not ”learn” from the data, as instead, it searches the database for the closest samples based on the feature similarity, and thus cannot be considered as ML. Moreover, the performance of $k$-NN is negatively affected by increasing the dataset size, especially in terms of positioning time and effort.

In this section, the sample-wise compression methods are introduced, discussed, proposed, and evaluated. Implementing a sample-wise radio map reduction method can be done using a heuristic [137], an outlier detection mechanism [15], [18], [138], or by transforming multiple samples based on their common aspects to a new representation, similar as in feature-wise compression.

The Author’s collaborative work [15] proposes an augmented DBSCAN method to remove outliers from the radio map based on the similarity across the samples’ features. Similarly, [18] introduces a sample reduction mechanism based on the correlation among training features. Both works boost positioning performance in terms

of the positioning errors $\varepsilon_{3D}$ and $\varepsilon_{2D}$, as well as improve the positioning estimation speed of the $k$-NN algorithm by decreasing the size of the training dataset. An iF_Ensemble method, proposed in [138], combines Support Vector Machine (SVM), $k$-NN, and Random Forest models to detect outliers with positive results. Clustering can also be utilized to cluster the samples and represent the number of samples based purely on their cluster features or consider only the samples within a relevant cluster to perform inference on [139]. The methods mentioned above reduce the number of samples based purely on their features and the relations between them.

The spatial filtering to reduce the number of considered samples is proposed in [122] based on the Hamming distance between vectors as a match in detected APs. The same work proposes a kernelized distance calculation between the measured RSS array and the training database, showing improved performance over generic $k$-NN.

The works presented above consist of mechanisms that compute similarity across samples, based on which they remove a subset of samples from the training database (either in pre-processing or as a part of the positioning scheme), effectively removing a piece of information from the training dataset.

In comparison, the novel method for sample-wise compression proposed in this thesis exploits the common features from all samples within the radio map based on their similarity in terms of labels. The indoor positioning datasets often contain multiple measurements per location, as denoted by $\mu_{S_{train}loc}$ in Table 3.3, where 17 out of 26 (disregarding TUT 4 with $\mu_{S_{train}loc} = 1.03$) of the considered datasets contain more than a single measurement per location, on average. The goal of the proposed dimensionality reduction scheme is to find an optimal representation for all samples sharing the same label (location).

In this section, we propose a novel dimensionality reduction scheme denoted radio Map compression Employing Signal Statistics (MESS), which consists of the radio map compression scheme, as well as of modification for a $k$-NN distance metric calculation, as described in the paragraphs below.

### 3.4.1 MESS: Method Derivation and Algorithm

The idea behind radio Map compression Employing Signal Statistics (MESS) is to represent each location in the training dataset using a single sample. Consequently, the compression is performed by extracting location-specific features from the set of

radio map elements that share the same label. The selection of relevant features can be limited to a simple AP-wise average, yet by implementing this approach a significant portion of the information can be lost, including the location-specific uncertainty of the measured quantity.

In order to represent the uncertainty of the measured RSS (or any other quantity used as a feature), the proposed MESS algorithm computes the Standard Deviation (SD) for each AP as the representative of the certainty, and therefore trustworthiness, of that feature. Consequently, the MESS algorithm transforms a set of training fingerprints sharing the same label into a single fingerprint that represents each original feature across all relevant samples using only two numbers. MESS only becomes efficient in case the mean number of training samples per location $\mu_{S_{train}/loc}$ introduced in Table 3.3 is greater than 2. In the scope of this work, we consider 13 positioning datasets with $\mu_{S_{train}/loc} \geq 5$, determining a relevant CR of MESS $CR_{MESS}$ as 2.5 or more as:

$$CR_{MESS} = \frac{\mu_{S_{train}/loc}}{2} \tag{3.18}$$

The CR is therefore constant for each dataset and does not depend on any other input parameters. The compression algorithm is implemented as follows:

1. Load the training dataset including the radio map and the corresponding labels.
2. Obtain the set of unique training locations in terms of $x$, $y$, and $z$ coordinates (the floor and building labels are accounted for, as each dataset shares the same coordinate system across the whole deployment.
3. For each unique location perform the following steps:
   (a) Find all samples with the corresponding location.
   (b) Compute the mean RSS for each AP, considering only the detected measurements (not accounting for the non-measured values).
   (c) Compute the SD of each AP's measurements, considering only the detected measurements. In case the SD equals 0 (single valid measurement or all measurements the same), 0 is stored.
   (d) Construct the reduced fingerprint as an array, where the element representing each AP consists of the mean and the SD pair (for convenience, the implementation considers the pair as $[\mu + i \cdot \sigma]$, where $\mu$ represents the mean, $\sigma$ denotes the SD and $i$ is the unit imaginary number).

(e) Add the reduced fingerprint with the corresponding label to the reduced dataset.

The compression scheme considers only the training part of each dataset, while the testing part can be processed by the positioning model without prior processing related to the compression method. Nevertheless, since each training radio map's element is now represented by two numbers instead of one, the distance metric that computes the similarity measure between the testing sample and all samples in the training database needs to be adjusted by taking into account the SDs as well.

Before matching, the algorithm calculates an array of the expected AP-specific SDs $E_{SD}$, which is utilized in the event of matching the testing sample element to a fingerprint with SD of the corresponding AP equal to 0.

For each AP, the expected SD $E_{SD}(a)$ is determined as a mean across the non-zero SDs from the training set. In case all training samples' SDs were zeros, the assigned expected SD was determined by a maximum SD from the whole training set.

The distance estimation between the training sample $p$, whose real part denotes $\mu$ and its imaginary part the $\sigma$ for each AP, and the testing sample $q$ is then performed in the following fashion for Sørensen distance augmentation for MESS method (denoted as MESSy Sørensen distance), as Sørensen distance is the benchmark distance considered in this work:

1. Calculate the element-wise (AP-wise) errors considering the real part of each feature.
2. Multiply the errors by an inverse of the relevant SDs, where the SD is the imaginary part of each training sample's feature. In case the received SD is 0, apply the expected SD as:

$$SDs(a) = \begin{cases} imag(p(a)), & \text{if } imag(p(a)) \neq 0 \\ E_{SD}(a), & \text{otherwise} \end{cases} \tag{3.19}$$

3. Apply the MESS power function to each element using the pre-defined exponent $e_{MESS}$. The choice of the value of $e_{MESS}$ is further discussed below.
4. Calculate the element-wise (AP-wise) contribution using the real part of each feature.
5. Multiply each contribution using the corresponding SD.

6. Apply the MESS power function to each contribution element.
7. Divide the sum of the resulting element-wise errors by the sum of the element-wise contributions.

Mathematically, MESSy Sørensen distance $dist_{Sorensen,MESS}$ is obtained as:

$$dist_{Sorensen,MESS}(f_i, \overline{f}_i) = \frac{\sum_{a=1}^{|AP|} (|f_{i,a} - \overline{f}_{i,a}| \cdot \sigma_{i,a})^{e_{MESS}}}{\sum_{a=1}^{|AP|} [(f_{i,a} + \overline{f}_{i,a}) \cdot \sigma_{i,a}]^{e_{MESS}}} \tag{3.20}$$

where $f_{i,a}$ denotes the real part of the corresponding training feature, $\sigma_{i,a}$ denotes the imaginary part of the corresponding training feature, and $e_{MESS}$ is the tunable exponent parameter.

Equation 3.20 introduces a tunable parameter $e_{MESS}$ to the proposed scheme, which can be interpreted as the error regularizer. Overall, the multiplication with the inverse of the SD weights each error by its relevant magnitude, relevant to the spread of values within the original samples. Consequently, higher errors can be diminished in case the distribution of the original samples was spread (high SD). The parameter $e_{MESS}$ further scales the obtained ratio depending on its value, increasing the impact of higher error ratios with increasing $e_{MESS}$.

The goal of the compression scheme is to capture and apply the effect of the signal strength fluctuations at each training location and then account for its effect within the distance calculation of the $k$-NN method. Consequently, each measurement is scaled by its expected fluctuation (SD), reducing the impact of the uncertain results and increasing the impact of the stable measurements acquired at the given location.

The algorithm of the proposed MESS compression is depicted in Figure 3.19.

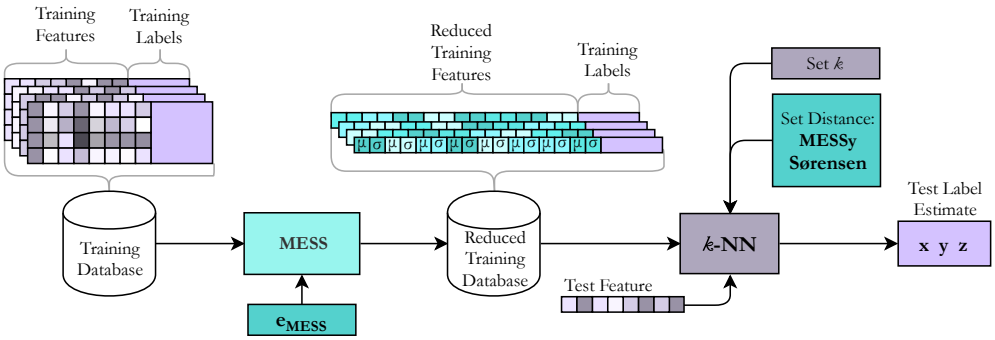


**Figure 3.19** MESS algorithm

### 3.4.2 Numerical Evaluation

This section evaluates the performance of the proposed MESS compression scheme in terms of the considered performance metrics. The provided evaluation investigates the effect of the $e_{MESS}$ parameter, plots the achieved positioning performance based on the varying number of considered samples $k$ in $k$-NN, and provides the numerical results for the best performing $e_{MESS}$ setting for each database.

The achieved positioning performance across the considered datasets and varying $e_{MESS}$ magnitude is presented in Figure 3.20 in terms of normalized 3D positioning error $\tilde{\varepsilon}_{3D\alpha}$ normalized toward the baseline $\alpha$. The results show that each dataset performs the best at different $e_{MESS}$ values, yet the lowest aggregated error across considered datasets was achieved at $e_{MESS} = 0.7$, and with increasing $e_{MESS}$ the performance deteriorates.

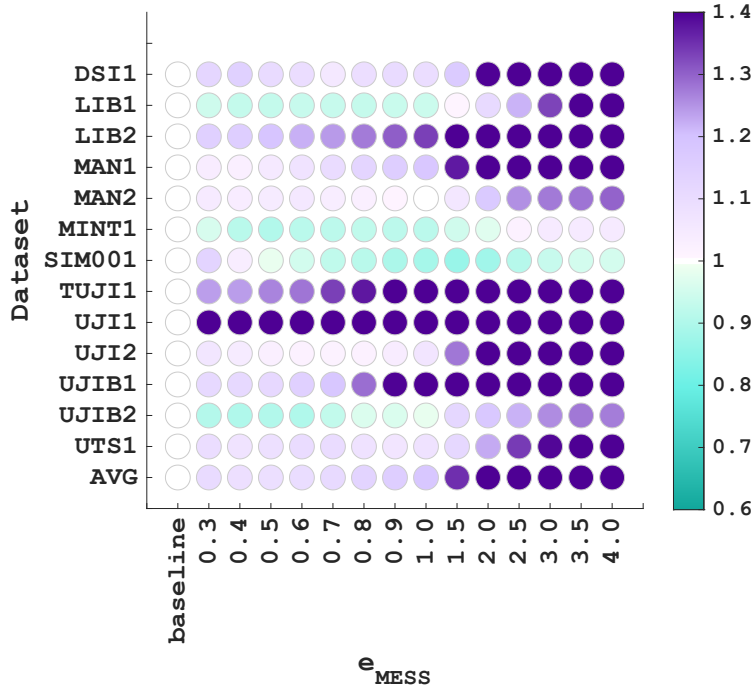


**Figure 3.20** Visualization of achieved positioning performance $\tilde{\varepsilon}_{3D\alpha}$ based on the parameter $e_{MESS}$.

The results in Figure 3.20 show that the MESS compression is capable of significant performance improvement in a considerable number of datasets, while in

some scenarios (TUJI 1, UJI 1), the method underperforms. The rationalization of the poor performance on the given datasets is that the uncertainties captured within the SD do not refer to the uncertainties of the radio measurements themselves. The TUJI 1 dataset was measured using 5 different devices with different parameters, thus the inconsistencies within the measurements are caused by different physical attributes of the devices much more than the variance of the signal strength itself.

The best-case performance of the MESS algorithm on each dataset is presented in Table 3.10, including the achieved CR, $e_{MESS}$ parameter setting and the normalized metrics towards the baseline $\alpha$. The aggregated results show that on average, the MESS algorithm increases the positioning error by 6%, yet the aggregated values do not represent the individual performance of the datasets.

**Table 3.10** Results of MESS with best exponent setting

| Dataset | $CR$ | $e_{MESS}$ | $\tilde{\zeta}_{\mathcal{B}\alpha}$ [-] | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] |
|---|---|---|---|---|---|---|---|
| DSI 1 | 2.98 | 0.7 | 1 | 1 | 1.06 | 1.06 | 0.23 |
| LIB 1 | 6 | 0.4 | 1 | 1 | 0.93 | 0.93 | 0.23 |
| LIB 2 | 6 | 0.3 | 1 | 1 | 1.15 | 1.13 | 0.21 |
| MAN 1 | 55 | 0.4 | 1 | 1 | 1.03 | 1.03 | 0.01 |
| MAN 2 | 5 | 1 | 1 | 1 | 1.00 | 1.00 | 0.12 |
| MINT 1 | 13.16 | 0.5 | 1 | 1 | 0.90 | 0.90 | 0.02 |
| SIM 1 | 5 | 1.5 | 1 | 1 | 0.87 | 0.87 | 0.06 |
| TUJI 1 | 2.5 | 0.3 | 1 | 1 | 1.24 | 1.24 | 0.23 |
| UJI 1 | 10.65 | 0.4 | 1 | 0.83 | 1.53 | 1.42 | 0.64 |
| UJI 2 | 5.26 | 0.7 | 1 | 1 | 1.02 | 1.06 | 1.41 |
| UJIB 1 | 15.25 | 0.4 | 1 | 1 | 1.11 | 1.11 | 0.03 |
| UJIB 2 | 12 | 0.4 | 1 | 1 | 0.90 | 0.90 | 0.04 |
| UTS 1 | 3.11 | 0.9 | 1 | 0.98 | 1.07 | 1.07 | 2.38 |
| Average | | | 1.00 | 0.99 | 1.06 | 1.06 | 0.43 |

The largest CR of 55 was achieved on the dataset MAN 1, which includes 110 measurements per location. The performance deterioration of 3% is accompanied by the prediction time reduction of 99%. Similarly, the MINT 1 dataset size was reduced by 13.16-fold while reducing the positioning error by 10% and the prediction time by

98%. The performance degradation was caused mainly by the two datasets on which the MESS algorithm had the most severe negative impact in terms of positioning performance, namely TUJI 1 and UJI 1. If removing the two datasets from the evaluation, the average shift in positioning performance disappears, as the aggregated normalized 2D and 3D positioning errors are 1.00.

The time required for positioning $\tau$ has considerably decreased for all but two datasets, namely UJI 2 and UTS 1, as the aggregated normalized positioning time in comparison to the baseline $\alpha$ $\tilde{\tau}_\alpha$ is 0.43. If not considering these two datasets, the aggregated normalized positioning time $\tilde{\tau}_\alpha$ further decreases to 0.17. The decrease in the positioning time is mainly caused by a smaller number of samples required by the positioning model to compare with the testing one by calculating the distance. The increased positioning time in datasets UJI 2 and UTS 1 is attributed to the higher complexity of the distance calculation across multiple APs, not compensating for the relatively low CRs.

The following evaluation considers evaluating the MESS algorithm on the varying number of considered neighbors $k$. Sweeping over a large number of parameters is one of the most tedious aspects of optimizing a fingerprinting model for the indoor scenario, where considering different distance metrics, varying numbers of neighbors $k$, or $k$-NN alternatives have to be considered. The proposed MESS algorithm reduces the search space on which the $k$ of $k$-NN has to be considered due to characterizing each training location directly. The effect is shown in Figure 3.21, visualizing the positioning performance of three iterations of MESS algorithm with $e_{MESS}$ = 0.5, 0.7, and 1 based on varying $k$ of the $k$-NN algorithm. The remaining parameters correspond to $\alpha$ baseline. The positioning performance on the original data with the $\alpha$ $k$-NN parameters, apart from varying $k$, is included for comparison.

The results on the LIB 1 dataset (Figure 3.21, top, left) show that the MESS algorithm achieves the optimal performance at $k$ = 3, outperforming the benchmark, despite achieving 6-fold compression. The exponent parameter has minimal effect on the positioning performance. The results achieved on the MINT 1 dataset show that the least positioning error was obtained at $k$ = 5 with $e_{MESS}$ = 1, outperforming the benchmark, while the evaluation on the TUJI 1 dataset shows that the MESS algorithm was not able to improve the positioning performance. The visualization of the UJIB 2 dataset shows the improvement of the positioning performance over the benchmark at $k$ smaller than 12, with optimum at $k$ = 7.

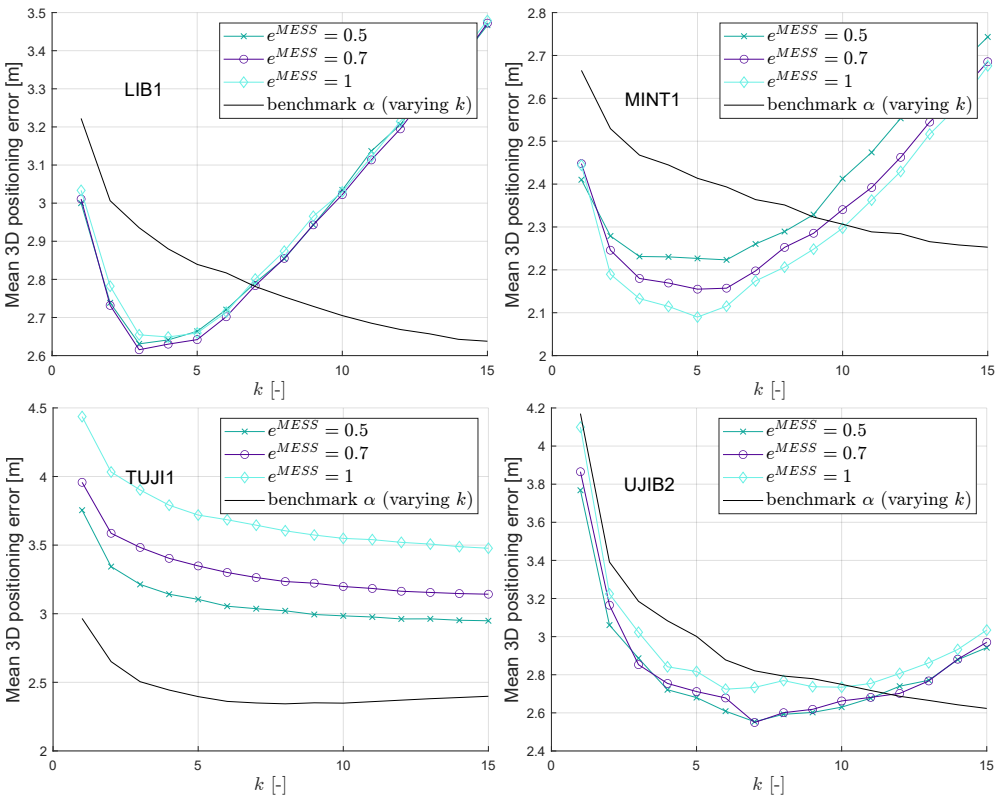


**Figure 3.21** Evaluation of MESS algorithm's positioning performance against the benchmark solution on four selected databases

The provided figures demonstrate the following impact of MESS algorithm on the behavior of $k$-NN:

- Due to the fact that each training sample represents a unique location, the optimal $k$ in $k$-NN is significantly limited. Specifically, the best-performing $k$ after applying the MESS compression will most likely be 3 or slightly larger, based on the label uncertainty ($k = 3$ interpolates between 3 closest training locations).

- In case the location-specific uncertainties do not represent the signal strength variance, the positioning error achieved while using the MESS algorithm does not converge at low $k$ values, as shown in TUJI 1 evaluation.

## 3.5 Concluding Remarks

In this chapter, various radio map compression schemes are proposed, implementing dimensionality reduction tasks across three different dimensions of the radio map to reduce the storage requirements, as well as the computational complexity of the fingerprinting indoor positioning system.

The bit-level compression scheme proposed by the Author of this thesis denoted Element-Wise cOmpression using $K$-means (EWO$K$) is introduced. EWO$K$ performs the compression via element-wise $K$-means clustering while removing the randomness from the initialization process. The numerical results show that the degradation of the positioning performance in terms of 3D positioning error and the floor hit is negligible across the 26 considered datasets, while in individual cases boosts the achievable positioning accuracy. The CR of EWO$K$ is calculated based on the 7-bit benchmark, yet in numerous cases, the original radio map is stored in considerably higher-bit format (integer, float), strongly increasing the resulting effective dimensionality reduction achieved by EWO$K$, e.g. using Int16 format as a baseline changes the CR achieved using the EWO$K$ with $K = 16$ to $16/4 = 4$. Moreover, EWO$K$ shows remarkable stability across all considered datasets, which, as shown in this chapter, is not often applicable.

The feature-wise compression was realized by applying Principal Component Analysis (PCA) decomposition as a compression mechanism, as well as by implementing the Auto-Encoder (AE), an unsupervised NN model. The radio map features were transformed using the corresponding models, limiting the achieved CR by sustaining the pre-defined explained variance within the data. The achieved results indicate that removing the redundancies from the radio map, caused by the sparse RSS measurements per fingerprint, is an efficient way to reduce the dataset size. The deterministic PCA performs Eigendecomposition, sustaining the information within the individual fingerprints, thus, providing excellent features for the $k$-NN while drastically reducing their number. Alternatively, AE demonstrates the generalization capabilities of the NN structures by iteratively adapting its parameters to the data in the training process. The numerical results show that feature-wise compression schemes achieve substantial CRs at the cost of a decrease in the positioning performance.

In the scope of sample-wise compression schemes, several heuristics and clustering-

based techniques were introduced in the literature, yet a label-oriented compression is, to the best knowledge of the Author, a novel idea. The compression scheme called radio Map compression Employing Signal Statistics (MESS) was proposed, which reduces the number of training samples by grouping them based on the matching labels, representing them using a single fingerprint. The achieved compression is determined by the number of samples in each dataset that share the same location, which varies from a single sample to 110 samples in dataset MAN 1. The $k$-NN similarity calculation had to be adjusted as both mean $\mu$ and SD $\sigma$ of the AP-wise elements are used to describe the reduced fingerprint. The provided evaluation shows that across the majority of the considered datasets, substantial compression was achieved, while in many cases improving the positioning performance. The MESS scheme can be tuned using the exponent parameter $e_{MESS}$ to optimize fitting each dataset. The disadvantage of the MESS algorithm is its weakness to other sources of uncertainty than the location-specific variance of the radio signal strength measurements.

Concluding this chapter, when implementing a radio map compression within a positioning scheme there are numerous variables that have to be considered before choosing the most-fitting option. Bit-level compression scheme EWO$K$ can be freely applied due to their reliability and the capability of incorporating new training samples into the reduced radio map. The downside is the limited CR the method is capable of achieving. A feature-wise compression, such as PCA, offers higher compression capabilities at the price of lower positioning performance. In case the NN is considered, careful evaluation, hyperparameter tuning, and validation need to be performed. The third option is to implement a sample-wise compression, that either removes samples classified as outliers from the training data or applies a data-driven transformation, such as the proposed MESS. The advantage is the possible compression and the performance improvement, while the downside of implementing the MESS algorithm is the requirement of multiple measurements per single location within the training data and a chance the method will underperform.

## 3.6 Author's Contributions

The main Author's contributions to the field of positioning and data compression discussed in this chapter may be summarized as follows:

- Introducing the dimensions of the radio map that can be reduced and incorpo-

rating a lossy compression scheme into the fingerprinting positioning scheme with $k$-NN model as the localization algorithm.

- Summarizing the relevant background of fingerprinting-based positioning schemes and their individual components.
- Describing and implementing relevant performance metrics to evaluate the effects of the compression scheme on the resulting positioning performance.
- Characterizing and evaluating a TUJI 1 indoor positioning dataset gathered and processed by the Author, which shall be soon available online.
- Introducing 25 additional indoor positioning datasets, that are openly available online and used to evaluate the performance of the proposed solutions.
- Establishing 2 relevant benchmarks for each dataset to mark the achievable positioning performance in a universal scenario and with fine-tuned parameters.
- Proposing Element-Wise cOmpression using $K$-means (EWO$K$), a bit-level radio map compression scheme based on element-wise clustering of the RSS values. Within EWO$K$ scheme, removing the effect of random initialization of the $K$-means clustering method by introducing a distribution-based algorithm to determine the initial cluster values that ensure the stability of the positioning system with optimal performance.
- Designing a centroid-update scheme to ensure unbiased reduced database updates within EWO$K$ compression scheme.
- Numerically evaluating the positioning performance of the EWO$K$ model, showing its scalable compression capabilities and minimum to no deterioration of the positioning performance across all considered datasets on both baseline settings.
- Introducing feature-wise radio map compression schemes, that enable significant dimensionality reduction due to the sparsity of the features within the radio map. As the representatives, Principal Component Analysis (PCA) and Auto-Encoder (AE) are introduced and implemented, while discussing different AE architectures and capabilities.
- Numerically evaluating both PCA and AE on the considered datasets provides a substantial reduction in dataset size and in the positioning time. The resulting performance deterioration of the PCA method is lower than that of the AE.

- Describing the different ways to perform a sample-wise radio map reduction within the SotA and introducing novel sample-wise compression scheme denoted radio Map compression Employing Signal Statistics (MESS) that capitalizes on available AP-wise statistics. MESS reduces the radio map by aggregating the samples with identical labels.
- Proposing new $k$-NN's distance metric calculation to incorporate the signal statistics derived during the MESS compression.
- Presenting the numerical evaluation of the MESS algorithm, showing the data-dependent performance with a large potential for performance improvement despite the substantial dimensionality reduction, while stabilizing the $k$ parameter, reducing the parameter sweeping requirements.
- Discussing the relevant aspects of each proposed compression scheme.

CHAPTER 4

# BOOSTING WEARABLE PERFORMANCE TO ENABLE ENERGY-EFFICIENT COMPUTING

Ensuring seamless, accurate and energy-efficient positioning is one of the main enablers, as well as the requirements of the 5G and beyond networks. The models proposed in this chapter aim to reduce the computational and storage requirements of the system by creating a cascaded system with best-performing structures for given tasks or by compressing the data utilizing multiple-stage compressions. Firstly, a method of combining multiple models and techniques to achieve such goals in challenging indoor environments is proposed. Secondly, this thesis elaborates on how can the radio map reduction methods introduced in Chapter 3 be effectively combined to achieve deep compression, reducing the size of the radio map while speeding up the $k$-NNs positioning estimation.

## 4.1 Accurate Positioning with Reduced Estimation Time

As discussed in Chapter 3, positioning is a crucial and inseparable capability of modern devices. Fast and precise positioning is a must, in terms of reliable and high-performing solutions in areas such as security, eHealth, and monitoring.

*k*-NN is considered one of the best-performing methods in terms of indoor positioning fingerprinting, as it achieves hard-to-match accuracy, it does not require any training, and it is straightforward to implement, but its main drawback remains unsolved - the lengthy execution time. On the one hand, the time required for the positioning of each of the testing fingerprints increases with the size of the database. On the other hand, the bigger the training dataset is, the better positioning accuracy is usually achievable.

The topic of the computational complexity of positioning was already addressed in the SotA literature as several works addressing the positioning time and complexity reductions were published. In [137] the authors propose three variants of *K*-means clustering of the radio map, one of which achieves up to 40% faster positioning performance compared to the traditional stand-alone *k*-NN fingerprinting. These results were obtained using 16 out of 26 datasets used in this thesis.

Boosting the positioning performance by cascading the models was realized by one of the teams at the "Data Analytics and ML" program's challenge [140]. The proposed cascade achieved the best results among student teams, on top of which, outperforming some of the benchmark solutions. Their results prove that implementing a series of models, rather than a singular solution, can positively impact positioning accuracy. Pre-clustering the fingerprints to reduce the search space was already mentioned in [139]. A two-stage cascading scheme, proposed in [141], first clusters the fingerprints using the *K*-means model, after which the classification model estimates the user location. Across the evaluated models, Decision Tree (DT) achieved the highest positioning accuracy in the considered small-scale deployment. A work proposing a cascade correlation neural network model [142] combines two NNs in a series to perform localization and tracking of an asset. The single-room evaluation shows promising results. Cascading multiple NNs was realized in [143], where the models in a sequence partitioned the deployment. The results show that

despite the mean error being reduced, the misclassified samples caused exceedingly high outliers in terms of positioning error.

In this section, a sequential search space reduction scheme is implemented by proposing a cascade of models. Although the idea of cascading the models itself was already studied, there are still inconsistencies and open questions that remain unanswered. Compared to the literature introduced above, the main novelty of the proposed solution lies in the exploration and analysis of the suitable models for each model within the cascade, as well as in the robust and thorough evaluation of such models. The section provides valuable insights and guidelines for designing multi-level positioning models. The idea was first published by the Author in [17] and achieves up to 100-fold reduction of time required for positioning in comparison to the $k$-NN $\alpha$ benchmark.

### 4.1.1 Method Derivation and Algorithm

The idea behind this cascading model is to find the best-performing mechanism for given tasks (building and floor classifiers, 2D regressor) while iteratively reducing the search space sizes. This consequently leads to faster positioning of the user as well as reduced computational requirements and a more energy-efficient system. It was first introduced by the Author in [17].

Apart from performing the positioning function, the task of the proposed model involves reducing the search space by dividing the original dataset according to the data labels into smaller datasets of included buildings, and their individual floors. Only after the search space is reduced, algorithms evaluated as the best-performing ones (during the so-called validation phase) are trained as follows:

During the **offline phase**, a building classifier is trained on the whole original training dataset so that it can predict the correct building label. The building classifier is followed by a floor classifier, which is trained only on the data with the given building label so that it can predict the correct floor, which is followed by a 2D regressor, which is trained to determine the actual user position using only the data corresponding to the given floor.

Later, during the **online phase**, the user position is estimated from the incoming measurement (here represented as a test feature) in terms of the building label $\mathcal{B}_i$, floor label $\mathcal{F}_i$, and finally the position in $x$, $y$, and $z$ coordinate, separately. The newly

measured fingerprint is processed by the pre-trained building classifier, determines the building label $\mathcal{B}_i$, stores it as a part of the final coordinate, and gives the label information to the system. Next, the floor classifier corresponding to one of the estimated building labels processes the fingerprint and estimates the floor label $\mathcal{F}_i$. Similar to the previous step, the floor label is stored and the estimate is used in order to choose only the respective subset of the training data to make the final $x$, $y$, and $z$ estimate. Figure 4.1 depicts this process.

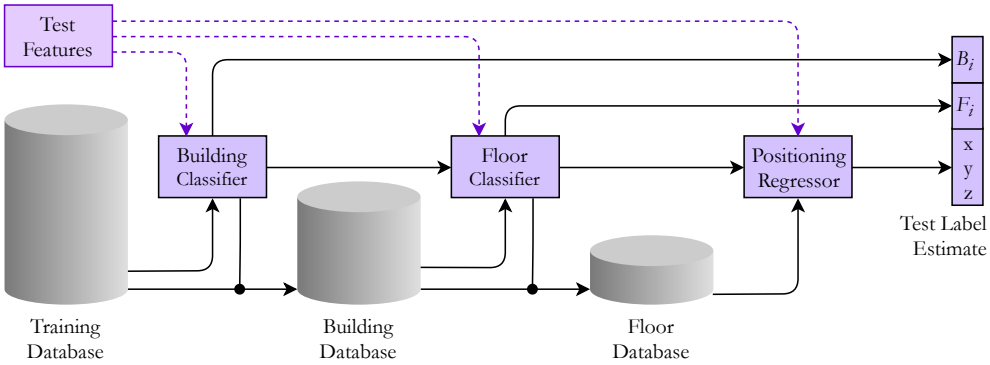


**Figure 4.1** Cascading - General idea

In [17] the Author evaluated an array of the ML models as a part of the validation phase, along with several configurations of $k$-NN to perform each of the three tasks within the system, and several relevant alternatives for the cascade were found. In the scope of this thesis, rather than repeating the sweep, a fine-tuning evaluation is performed to improve the performance of each model in terms of computational complexity and positioning capabilities.

In terms of the building classification model, a neural network with 100 neurons in a hidden layer was found as the optimal solution, while a simple solution, such as the Naive Bayes classifier, achieved only slightly worse results. The optimization target of the building classification model is to reduce the NN model to a necessary minimum in order to sustain favorable performance while minimizing the complexity of the model.

The floor-hit estimator determined in [17] as optimal was a NN as well (with the same architecture), which achieved 99.2% floor hit accuracy on the validation set and improved the $\zeta_{\mathcal{F}}$ by 4% in later evaluation.

Finally, the best-performing positioning regressor was found to be the $k$-NN with $k = 1$, outperforming the other considered estimators by a relevant margin.

An additional aspect to consider when implementing a positioning model is the size of the model itself. When considering $k$-NN, the model is comprised of several lines of code, and the required data volume is the training database itself. The NN architecture can be stored as a set of weights and the corresponding architecture in a relatively lightweight form as well. Meanwhile, the physical size of the decision tree-based classifiers or regressors is considerably larger, especially in case an ensemble of models is considered, as each decision boundary within the tolerable depth of each tree has to be defined and stored.

### 4.1.2 Numerical Evaluation

In this section, the numerical results concerning the cascading method are provided as well as the results of the validation phase, which is used in order to determine the best combination of classifiers and regressors to be used. The datasets considered in this section include only those from Table 3.3, which include more than one floor. In total, 14 datasets were chosen, two of which consist not only of multiple floors, but also multiple buildings.

#### Validation

In order to determine the best-performing combination of classifiers and a regressor to be used in the cascade, a validation phase is carried out. It consists of a performance evaluation of considered methods (in their pre-defined settings) using a small subset of the training data as a validation set and choosing the combination based on the accuracy and execution times. In the scope of the experiment carried out for this thesis, 20% of the original training data were used as a validation set (see Table 3.3). The validation data were randomly picked from the training data so that there were no samples with the same label present in both the validation and (remaining) testing sets.

Models considered in the validation phase include:

- **1NN - $k$-Nearest Neighbors** with $k = 1$ and Sørensen distance (the same setting as benchmark $\alpha$ introduced in Section 3.1).
- **W3NN - Weighted $k$-Nearest Neighbors**with $k = 3$ and Sørensen distance. The Weighted $k$-NN is an alternative of $k$-NN, which linearly weights each

selected neighbor by its distance (the more similar the neighbor, the larger weight it has).

- **DT - Decision Tree:** A ML construct that creates a multitude of decision boundaries, shaping a tree-like structure. The selected parameters of the model include Gini impurity with greedy splitting mechanics and a maximum depth of 100.
- **RFor - Random Forest:** A model that creates a number of randomized decision trees and ensembles their outputs to form the final estimate. The considered parameters include 50 trees, the maximum depth of each tree as 50, and considers a square root of features at each tree.
- **AdB - AdaBoost:** Similar model to Random Forest (RFor) [144], creating a larger number of shallow trees that iteratively improve performance of the misclassified samples. In this thesis, AdaBoost (AdB) includes 200 trees with a depth of 1.
- **NN$_{simple}$ - Neural Network:** The simplest NN architecture that connects the inputs directly to the outputs. All neural networks consider Adam optimizer [145] and are trained for a maximum of 300 epochs with an early stopping mechanism with the patience of 10.
- **NN$_{10}$ - Neural Network:** A NN with 10 neurons in the intermediate layer with ReLU activation function.
- **NN$_{100}$ - Neural Network:** A NN with 100 neurons in the intermediate layer.
- **NN$_{[100,100]}$ - Neural Network:** A NN with two intermediate layers, each with 100 neurons.

All considered models are implemented in Python 3.6 where the models were generated and evaluated within the functions of the Scikit-Learn library. All considered models were designed as classifiers and regressors with the same parameters. Before applying the model to the data, the datasets were pre-processed by applying the positive data representation. In contrast to the evaluation within the Author's prior work [17], the models utilized in this thesis are restricted to the best-performing ones and aim to optimize them further.

### Validation set evaluation

To derive the best-performing model for each scenario, the validation procedure was first performed while splitting the training dataset into training and validation parts in an 80-20 ratio for each scenario. Based on the visualization in Figure 4.1, the building classifiers were assessed while training on the training data and evaluating the validation samples. The datasets with more than one building were then split based on the corresponding building and the floor classifier models were assessed with the corresponding split. Similarly, the positioning regressors were evaluated on the data that belong to the corresponding floor only.

The positioning results on the validation data achieved by the considered building classifiers are shown in Table 4.1 for the classification accuracy and the validation set prediction times. The results show that 1NN, RFor, $NN_{100}$, and $NN_{[100,100]}$ classifiers achieved 100% floor hit rate, from which the $NN_{100}$ model performed the fastest and thus is considered as the go-to solution.

**Table 4.1** Building classification results - Validation

| Dataset | 1NN | W3NN | DT | RFor | AdB | $NN_{simple}$ | $NN_{10}$ | $NN_{100}$ | $NN_{[100,100]}$ |
|---|---|---|---|---|---|---|---|---|---|
| **Validation - Number of misclassified samples [-]** | | | | | | | | | |
| UJI 1 | 0 | 0 | 2 | 0 | 188 | 6 | 5 | 0 | 0 |
| UJI 2 | 0 | 2 | 3 | 0 | 38 | 11 | 8 | 0 | 0 |
| Average | 0 | 1 | 2.5 | 0 | 113 | 8.5 | 6.5 | 0 | 0 |
| **Validation - Prediction time [s]** | | | | | | | | | |
| UJI 1 | 37.17 | 36.48 | 0.002 | 0.029 | 0.926 | 0.010 | 0.008 | 0.014 | 0.018 |
| UJI 2 | 41.01 | 40.61 | 0.009 | 0.022 | 1.002 | 0.013 | 0.010 | 0.015 | 0.022 |
| Average | 39.09 | 38.55 | 0.006 | 0.025 | 0.964 | 0.012 | 0.009 | 0.015 | 0.020 |

Floor classifiers are evaluated in terms of accuracy in Table 4.2, and in terms of positioning time in Table 4.3. The results of the validation accuracy show highly comparable results across a number of classifiers, with $NN_{[100,100]}$ as the only model scoring higher than 99% accuracy on the validation sets, closely followed by $NN_{100}$ and 1NN models. Therefore, $NN_{[100,100]}$ is utilized further as the best-performing

method for floor classification.

**Table 4.2** Floor classification results - Validation accuracy [%]

| Dataset | 1NN | W3NN | DT | RFor | AdB | $NN_{simple}$ | $NN_{10}$ | $NN_{100}$ | $NN_{[100,100]}$ |
|---|---|---|---|---|---|---|---|---|---|
| LIB 1 | 100 | 100 | 99.14 | 100 | 100 | 98.28 | 100 | 99.14 | 100 |
| LIB 2 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 |
| SAH 1 | 100 | 100 | 99.73 | 100 | 95.05 | 100 | 100 | 100 | 99.95 |
| TIE 1 | 100 | 99.86 | 99.62 | 100 | 71.37 | 99.86 | 99.76 | 99.95 | 99.91 |
| TUT 1 | 97.97 | 98.31 | 92.57 | 98.65 | 65.54 | 98.31 | 98.99 | 98.99 | 98.99 |
| TUT 2 | 100 | 99.15 | 97.44 | 99.15 | 94.02 | 97.44 | 99.15 | 99.15 | 99.15 |
| TUT 3 | 89.29 | 82.86 | 73.57 | 89.29 | 47.86 | 91.43 | 92.14 | 92.86 | 95 |
| TUT 4 | 95.70 | 95.83 | 86.85 | 96.71 | 61.06 | 94.44 | 95.95 | 95.70 | 95.70 |
| TUT 5 | 100 | 100 | 93.33 | 94.44 | 61.11 | 100 | 98.89 | 98.89 | 100 |
| TUT 6 | 99.84 | 100 | 97.44 | 99.52 | 86.06 | 98.72 | 99.36 | 99.52 | 100 |
| TUT 7 | 98.92 | 98.57 | 96.24 | 98.92 | 93.73 | 98.39 | 98.75 | 99.10 | 98.75 |
| UJI $1_1$ | 99.81 | 99.71 | 97.71 | 99.90 | 77.71 | 99.43 | 98.95 | 99.14 | 99.24 |
| UJI $1_2$ | 99.90 | 99.81 | 98.64 | 99.90 | 68.51 | 99.52 | 99.52 | 99.61 | 99.71 |
| UJI $1_3$ | 99.84 | 99.79 | 97.78 | 100 | 57.06 | 99.37 | 99.42 | 99.58 | 99.58 |
| UJI $2_1$ | 98.70 | 98.96 | 95.68 | 99.65 | 74.42 | 98.70 | 98.53 | 98.96 | 98.62 |
| UJI $2_2$ | 99.09 | 98.81 | 96.16 | 98.99 | 65.36 | 99.09 | 98.81 | 99.09 | 98.99 |
| UJI $2_3$ | 99.64 | 99.43 | 97.33 | 99.85 | 68.95 | 99.23 | 99.23 | 99.33 | 99.49 |
| UTS 1 | 99.95 | 99.95 | 99.67 | 99.95 | 22.72 | 99.95 | 99.95 | 99.95 | 99.95 |
| Average | 98.81 | 98.39 | 95.49 | 98.61 | 72.81 | 98.45 | 98.74 | 98.83 | 99.06 |

In terms of validation prediction times, NN models sorted by their complexity, as well as DT performed the validation inference the fastest.

Since there are 82 individual floors over 18 buildings in 14 considered datasets, Table 4.4 provides the averaged results across all considered datasets' individual floors. The full tables containing all the results may be found in Appendix A in Table A.1 (validation accuracy) and Table A.2 (validation prediction times). Each dataset's results are provided per their respective buildings (denoted by the numbered index) and their individual floors (denoted by capital letters).

Table 4.4 shows that the best-performing positioning regression model is 1NN with a mean positioning error of 2.69 m across all floors, followed by W3NN model

**Table 4.3** Floor classification results - Validation prediction time [s]

| Dataset | 1NN | W3NN | DT | RFor | AdB | $NN_{simple}$ | $NN_{10}$ | $NN_{100}$ | $NN_{[100,100]}$ |
|---|---|---|---|---|---|---|---|---|---|
| LIB 1 | 0.012 | 0.009 | 0 | 0.010 | 0.051 | 0 | 0 | 0 | 0.001 |
| LIB 2 | 0.013 | 0.010 | 0 | 0.010 | 0.041 | 0 | 0 | 0 | 0 |
| SAH 1 | 12.562 | 12.030 | 0 | 0.018 | 0.699 | 0.008 | 0 | 0.010 | 0.002 |
| TIE 1 | 12.806 | 12.293 | 0.008 | 0.023 | 0.663 | 0.008 | 0.010 | 0.010 | 0.010 |
| TUT 1 | 0.113 | 0.108 | 0 | 0.002 | 0.053 | 0 | 0.008 | 0 | 0 |
| TUT 2 | 0.026 | 0.020 | 0 | 0 | 0.035 | 0 | 0 | 0 | 0 |
| TUT 3 | 0.115 | 0.091 | 0 | 0.016 | 0.063 | 0 | 0 | 0 | 0.002 |
| TUT 4 | 2.921 | 2.884 | 0 | 0.018 | 0.430 | 0 | 0 | 0 | 0.008 |
| TUT 5 | 0.018 | 0.016 | 0 | 0.010 | 0.048 | 0 | 0 | 0 | 0 |
| TUT 6 | 1.237 | 1.146 | 0.002 | 0.010 | 0.251 | 0 | 0 | 0.008 | 0.008 |
| TUT 7 | 1.204 | 1.139 | 0 | 0.008 | 0.254 | 0 | 0 | 0 | 0 |
| UJI $1_1$ | 2.591 | 2.553 | 0.002 | 0.010 | 0.284 | 0.001 | 0.002 | 0.008 | 0.004 |
| UJI $1_2$ | 2.447 | 2.427 | 0 | 0.018 | 0.286 | 0 | 0.002 | 0 | 0 |
| UJI $1_3$ | 8.218 | 8.214 | 0 | 0.020 | 0.512 | 0.002 | 0.008 | 0.008 | 0.010 |
| UJI $2_1$ | 3.108 | 3.075 | 0.008 | 0.018 | 0.368 | 0 | 0 | 0.002 | 0 |
| UJI $2_2$ | 2.747 | 2.733 | 0.002 | 0.012 | 0.309 | 0 | 0.002 | 0 | 0.002 |
| UJI $2_3$ | 8.893 | 8.686 | 0.008 | 0.021 | 0.552 | 0 | 0 | 0.002 | 0.011 |
| UTS 1 | 9.158 | 8.644 | 0.005 | 0.038 | 0.835 | 0.005 | 0.004 | 0.009 | 0.012 |
| Average | 3.788 | 3.671 | 0.002 | 0.015 | 0.319 | 0.001 | 0.002 | 0.003 | 0.004 |

**Table 4.4** 2D positioning regression results - Validation

| Dataset | 1NN | W3NN | DT | RFor | AdB | $NN_{simple}$ | $NN_{10}$ | $NN_{100}$ | $NN_{[100,100]}$ |
|---|---|---|---|---|---|---|---|---|---|
| **Validation Accuracy [m]** | | | | | | | | | |
| Average | 2.69 | 2.93 | 4.22 | 3.77 | 7.99 | 11.99 | 6.69 | 4.60 | 3.77 |
| **Validation Prediction Time [s]** | | | | | | | | | |
| Average | 0.221 | 0.221 | 0.002 | 0.019 | 0.089 | 0.000 | 0.001 | 0.001 | 0.001 |
| **Number of floor datasets with the lowest $\varepsilon_{2D}$ [-]** | | | | | | | | | |
| | 49 | 20 | 4 | 6 | 0 | 0 | 0 | 0 | 5 |

and $NN_{[100,100]}$. In terms of validation prediction times, NNs and DT perform the strongest. The table further shows the numerical analysis of the best-performing models per floor, where in 49 floors 1NN achieved the lowest positioning error, W3NN performed the best on 20 floors, while DT, RFor, and $NN_{[100,100]}$ showed dominating performance on several floors as well.

### Test set evaluation

Based on the validation, the two alternatives for combining the models are chosen. First, denoted the universal setting, is the sequence of models best fitting the majority of the models according to the validation. The chosen sequence of models includes $NN_{100}$ as the building classifier, $NN_{[100,100]}$ as the floor classifier, and 1NN model to perform the positioning regression.

The results of evaluating the universal cascade setting on the testing dataset are presented in Table 4.5, where the results achieved by the individual datasets, as well as the aggregated results, are presented while being normalized towards the baseline $\alpha$. On average, the cascade improves the building hit by 4%, damages the floor hit by 3%, and slightly decreases the 2D and 3D positioning performance.

The aggregated floor hit is devaluated by TIE 1 dataset, which correctly classified only a single sample across the testing set. If omitting this dataset from the evaluation, the aggregated floor hit increases to 1.04, and both 2D and 3D normalized positioning errors are improved to 0.99, meaning the overall performance of remaining datasets is improved in comparison to the benchmark $\alpha$.

Notably, the positioning time was improved by 82% on average when using cascade, when compared to the stand-alone $k$-NN with $\alpha$ parameter settings. The positioning time is reduced by up to 98% on datasets UJI 1, UJI 2, and UTS 1.

The second model alternative is to find the best-performing combination of models for each dataset separately based on its own validation performance. The model combinations, as well as the normalized results towards the baseline $\alpha$, are provided in Table 4.6. The best-performing models were chosen based on the majority vote in case different models were chosen across various floors or buildings.

The aggregated normalized results in Table 4.6 show marginally improved results compared to the universal cascade setting. The floor hit is improved by 4%, normalized 3D positioning error by 3%, and normalized 2D positioning error by 4%, with accelerated positioning time by 71% on average. Notably, there are 8 datasets with the

**Table 4.5** Numerical results of the cascade, universal setting

| | $NN_{100} \rightarrow NN_{[100,100]} \rightarrow 1NN$ | | | | |
|---|---|---|---|---|---|
| Dataset | $\tilde{\zeta}_{\mathcal{B}\alpha}$ [-] | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] |
| LIB 1 | - | 0.99 | 1.01 | 1.00 | 0.58 |
| LIB 2 | - | 0.91 | 1.14 | 1.06 | 0.57 |
| SAH 1 | - | 0.97 | 0.95 | 0.95 | 0.09 |
| TIE 1 | - | 0.04 | 1.35 | 1.29 | 0.04 |
| TUT 1 | - | 1.39 | 0.99 | 0.99 | 0.14 |
| TUT 2 | - | 1.09 | 0.96 | 0.96 | 0.37 |
| TUT 3 | - | 1.04 | 1.02 | 1.03 | 0.13 |
| TUT 4 | - | 1.01 | 1.02 | 1.03 | 0.06 |
| TUT 5 | - | 1.05 | 1.07 | 1.08 | 0.27 |
| TUT 6 | - | 1.00 | 1.00 | 1.00 | 0.13 |
| TUT 7 | - | 1.00 | 1.12 | 1.12 | 0.11 |
| UJI 1 | 1.01 | 0.98 | 0.99 | 0.99 | 0.02 |
| UJI 2 | 1.07 | 1.08 | 0.64 | 0.63 | 0.02 |
| UTS 1 | - | 1.03 | 1.00 | 1.04 | 0.02 |
| Average | 1.04 | 0.97 | 1.02 | 1.01 | 0.18 |

best-performing combination the same as the universal one, supporting the choice of the considered universal solution.

Based on the numerical evaluation and results provided in this section, implementing the cascading mechanism as described above may lead to a slight improvement in the positioning accuracy, but mainly leads to significant time reduction caused by the method's smaller search space and consequently a vast reduction of the number of performed operations, leading to the more energy-efficient system.

**Table 4.6** Numerical results of the cascade, adjusted setting

| Dataset | Cascade setting | $\tilde{\zeta}_{\mathcal{B}\alpha}$ [-] | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] |
|---|---|---|---|---|---|---|
| LIB 1 | - → $NN_{[100,100]}$ → 1NN | - | 0.99 | 1.01 | 1.00 | 0.53 |
| LIB 2 | - → $NN_{[100,100]}$ → 1NN | - | 0.91 | 1.14 | 1.06 | 0.57 |
| SAH 1 | - → $NN_{[100,100]}$ → 1NN | - | 0.97 | 0.95 | 0.95 | 0.09 |
| TIE 1 | - → 1NN → 1NN | - | 1.00 | 1.00 | 1.00 | 0.57 |
| TUT 1 | - → $NN_{10}$ → W3NN | - | 1.41 | 0.92 | 0.92 | 0.14 |
| TUT 2 | - → $NN_{[100,100]}$ → 1NN | - | 1.09 | 0.96 | 0.96 | 0.37 |
| TUT 3 | - → $NN_{[100,100]}$ → RFor | - | 1.04 | 0.93 | 0.94 | 1.49 |
| TUT 4 | - → RFor → RFor | - | 1.03 | 0.87 | 0.88 | 0.27 |
| TUT 5 | - → $NN_{[100,100]}$ → 1NN | - | 1.05 | 1.07 | 1.08 | 0.27 |
| TUT 6 | - → $NN_{[100,100]}$ → 1NN | - | 1.00 | 1.00 | 1.00 | 0.15 |
| TUT 7 | - → $NN_{100}$ → 1NN | - | 1.00 | 1.02 | 1.02 | 0.10 |
| UJI 1 | $NN_{100}$ → $NN_{[100,100]}$ → 1NN | 1.01 | 0.98 | 0.99 | 0.99 | 0.02 |
| UJI 2 | $NN_{100}$ → $NN_{100}$ → 1NN | 1.07 | 1.04 | 0.63 | 0.62 | 0.02 |
| UTS 1 | - → $NN_{[100,100]}$ → 1NN | - | 1.03 | 1.00 | 1.04 | 0.02 |
| Average | | 1.04 | 1.04 | 0.97 | 0.96 | 0.33 |

## 4.2 Integrating Lossy Compression Schemes for Effortless Localization

In Chapter 3, several radio map compression schemes were introduced, performing the dimensionality reduction task in three different dimensions. The results obtained on up to 26 different indoor positioning datasets show substantial size reduction, with only a minimal deterioration of the positioning capabilities on average, while in a number of cases improving the positioning performance. In this section, the combination of the individual compression schemes is evaluated, achieving multi-dimensional compression of the radio map, leading to drastically enhanced CRs due to the multiplicative nature of their combining, while, in many cases, also vastly reducing the time required for positioning.

The considered combinations of the methods are bit-level EWO$K$ with feature-wise PCA and AE due to a straightforward cascading of the methods, as explained in the following section, as well as the sample-wise MESS. The achieved performance gain in terms of reduced size and improved database processing scheme is evaluated

against the shift in the positioning performance achieved on the dataset when compared to the one achieved after the individual methods, as well as on the original dataset. The $k$-NN settings remain relevant to the baseline setting $\alpha$.

### 4.2.1 Sequencing the Proposed Compression Schemes

Sequencing the compression methods needs to be implemented in a consistent order to sustain the information contained in the data. First, the feature-wise compression scheme needs to be implemented, as it transforms the individual measurements within the radio map to a reduced representation, breaking the AP-wise organization of the radio map's features. Next, sample-wise compression follows, which extracts the common features from a multitude of samples, creating the common fingerprint characterizing the ones sharing the same label. Finally, the bit-level compression of the database using EWO$K$ is performed, reducing the resolution of the individual reduced radio map's elements. The whole process is depicted in Figure 4.2.

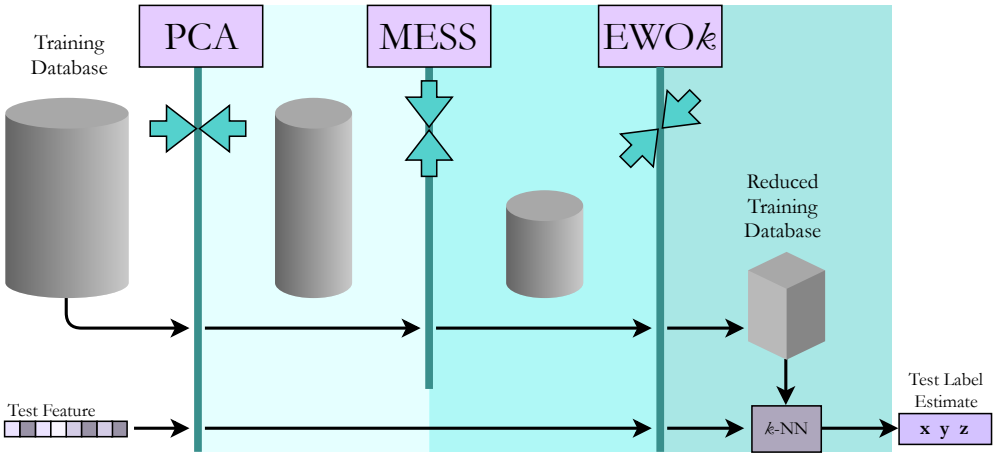


**Figure 4.2** Sequencing the proposed compression schemes

### 4.2.2 Numerical Evaluation

This section includes the exhaustive and final evaluation of combining the considered radio map compression schemes and their effect on positioning performance. The results involve the combination of feature-wise compressions PCA or AE (in-

troduced in Section 3.3) with novel bit-level EWO$K$ (introduced in Section 3.2), denoted feature-wise EWO$K$, the combination of sample-wise compression MESS (proposed in Section 3.4) with EWO$K$, denoted MESSy EWO$K$, and the combination of the three compression schemes compressing the radio map in all three dimensions, namely PCA, MESS, and EWO$K$, further referred to as Principal MESSy EWO$K$.

### Feature-wise EWO$K$

By combining the feature-wise compression scheme, either PCA or AE with bit-level EWO$K$, the compression is essentially achieved by discretizing the reduced features. The numerical results of the evaluation of feature-wise models at $\mathcal{T} = 90$ with EWO8 and initialization $\psi_{xtr}$ for all available datasets are presented in Table 4.7. The achieved combined CR ranges from 4.3 on dataset UJIB 2 to 101.5 achieved on LIB 1, averaging to considerable 47.5-fold size reduction across the datasets. The deep compression has in most cases negative impact on positioning accuracy, as the resulting $\tilde{\varepsilon}_{3D\alpha}$ equals 1.26 for PCA and 1.32 for AE implementation, while the sample positioning speed was increased by 89% and 92%, respectively. Notably, the implementation of EWO$K$ does not decrease the positioning time, as the matching is performed in standardized formats.

Table 4.8 shows the results of positioning and database reduction performance achieved by the feature-wise compression schemes with the EWO16 method. In terms of CR, the reduction is lower than in the previous case, while the positioning performance remains comparable in the majority of cases. This evaluation suggests that the combination with EWO8 may be more suitable for the majority of implementations.

Similarly, the combination of PCA or AE with EWO32 is evaluated in Table 4.9, where the performance in terms of positioning remains constant, while the CR has further decreased.

The visualization of the $\tilde{\varepsilon}_{3D\alpha}$ behavior based on the varying Threshold $\mathcal{T}$ is included in Figure 4.3. It shows the relevant trade-offs between the achieved compression ratios and the positioning errors for PCA, aggregated across all 26 datasets. The combinations of PCA with EWO$K$ clearly outperform the stand-alone PCA compression, while the best-performing combination in terms of the trade-off varies between PCA+EWO8 and PCA+EWO16.

**Table 4.7** Numerical results for PCA+EWO8 and AE+EWO8 at $\mathcal{T} = 90$

| | | PCA+EWO8 | | | | AE+EWO8 | | | |
|---|---|---|---|---|---|---|---|---|---|
| Dataset | CR | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] |
| DSI 1 | 21.5 | - | 1.33 | 1.33 | 0.10 | - | 1.30 | 1.30 | 0.12 |
| DSI 2 | 21.5 | - | 1.20 | 1.20 | 0.12 | - | 1.30 | 1.30 | 0.17 |
| LIB 1 | 101.5 | 1.00 | 0.98 | 0.99 | 0.04 | 0.98 | 1.17 | 1.14 | 0.03 |
| LIB 2 | 76.6 | 0.99 | 1.48 | 1.46 | 0.11 | 0.98 | 1.63 | 1.60 | 0.10 |
| MAN 1 | 13.1 | - | 1.20 | 1.20 | 0.03 | - | 1.60 | 1.60 | 0.03 |
| MAN 2 | 32.7 | - | 1.27 | 1.27 | 0.06 | - | 1.59 | 1.59 | 0.15 |
| MINT 1 | 12.8 | - | 1.15 | 1.15 | 0.05 | - | 1.02 | 1.02 | 0.04 |
| SAH 1 | 90.4 | 1.43 | 1.32 | 2.44 | 0.08 | 1.40 | 1.05 | 1.82 | 0.04 |
| SIM 1 | 6.2 | - | 0.89 | 0.89 | 0.03 | - | 1.23 | 1.23 | 0.04 |
| TIE 1 | 79.5 | 0.30 | 1.49 | 1.77 | 0.09 | 0.33 | 1.74 | 1.76 | 0.07 |
| TUJI 1 | 36.2 | - | 1.22 | 1.22 | 0.18 | - | 1.32 | 1.32 | 0.04 |
| TUT 1 | 48.1 | 0.97 | 1.21 | 1.19 | 0.09 | 0.92 | 1.25 | 1.13 | 0.05 |
| TUT 2 | 63.5 | 0.94 | 1.16 | 1.01 | 0.28 | 0.86 | 1.29 | 1.31 | 0.11 |
| TUT 3 | 74.7 | 0.99 | 1.13 | 1.05 | 0.12 | 0.98 | 1.13 | 1.07 | 0.12 |
| TUT 4 | 70.1 | 0.99 | 1.10 | 1.07 | 0.18 | 1.00 | 1.11 | 1.09 | 0.12 |
| TUT 5 | 81.5 | 0.97 | 1.23 | 1.23 | 0.08 | 0.99 | 1.27 | 1.28 | 0.10 |
| TUT 6 | 80.1 | 1.00 | 1.78 | 1.79 | 0.06 | 1.00 | 1.75 | 1.76 | 0.06 |
| TUT 7 | 85.0 | 0.99 | 1.66 | 1.68 | 0.05 | 0.99 | 1.47 | 1.49 | 0.05 |
| UEX B1 | 70.0 | 0.48 | 2.00 | 1.83 | 0.04 | 0.40 | 2.02 | 1.75 | 0 |
| UEX B2 | 35.0 | 0.72 | 1.50 | 1.36 | 0.08 | 0.79 | 1.52 | 1.42 | 0.08 |
| UEX B3 | 14.0 | 0.82 | 1.22 | 1.30 | 0.24 | 0.93 | 1.10 | 1.20 | 0 |
| UJI 1 | 36.8 | 0.98 | 1.05 | 1.07 | 0.15 | 0.99 | 1.12 | 1.11 | 0.08 |
| UJI 2 | 35.7 | 1.02 | 1.02 | 1.08 | 0.17 | 1.02 | 1.07 | 1.10 | 0.10 |
| UJIB 1 | 5.1 | - | 1.07 | 1.07 | 0.13 | - | 1.07 | 1.07 | 0.15 |
| UJIB 2 | 4.3 | - | 1.12 | 1.12 | 0.10 | - | 1.09 | 1.09 | 0.07 |
| UTS 1 | 40.4 | 1.04 | 1.09 | 1.09 | 0.20 | 1.01 | 1.07 | 1.07 | 0.13 |
| Average | 47.5 | 0.92 | 1.26 | 1.30 | 0.11 | 0.92 | 1.32 | 1.33 | 0.08 |

**Table 4.8** Numerical results for PCA+EWO16 and AE+EWO16 at $\mathcal{T} = 90$

| | | PCA+EWO16 | | | | AE+EWO16 | | | |
|---|---|---|---|---|---|---|---|---|---|
| Dataset | CR | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] |
| DSI 1 | 16.2 | - | 1.20 | 1.20 | 0.08 | - | 1.27 | 1.27 | 0.17 |
| DSI 2 | 16.2 | - | 1.17 | 1.17 | 0.17 | - | 1.31 | 1.31 | 0.21 |
| LIB 1 | 76.1 | 1.00 | 0.98 | 0.99 | 0.01 | 0.98 | 1.21 | 1.18 | 0.01 |
| LIB 2 | 57.5 | 0.99 | 1.49 | 1.46 | 0.08 | 0.97 | 1.64 | 1.58 | 0.11 |
| MAN 1 | 9.8 | - | 1.20 | 1.20 | 0.03 | - | 1.52 | 1.52 | 0.03 |
| MAN 2 | 24.5 | - | 1.16 | 1.16 | 0.04 | - | 1.50 | 1.50 | 0.06 |
| MINT 1 | 9.6 | - | 1.08 | 1.08 | 0.04 | - | 1.12 | 1.12 | 0.03 |
| SAH 1 | 67.8 | 1.29 | 1.17 | 2.00 | 0.07 | 1.35 | 1.04 | 1.99 | 0.08 |
| SIM 1 | 4.7 | - | 1.03 | 1.03 | 0.03 | - | 1.13 | 1.13 | 0.04 |
| TIE 1 | 59.6 | 0.12 | 1.38 | 1.49 | 0.10 | 0.09 | 1.82 | 1.51 | 0.04 |
| TUJI 1 | 27.1 | - | 1.18 | 1.18 | 0.18 | - | 1.29 | 1.29 | 0.06 |
| TUT 1 | 36.1 | 0.97 | 1.17 | 1.14 | 0.05 | 0.94 | 1.28 | 1.16 | 0.08 |
| TUT 2 | 47.7 | 0.93 | 1.09 | 1.00 | 0.14 | 0.88 | 1.18 | 1.17 | 0.23 |
| TUT 3 | 56.0 | 0.99 | 1.10 | 1.03 | 0.14 | 0.98 | 1.11 | 1.05 | 0.13 |
| TUT 4 | 52.6 | 1.00 | 1.09 | 1.07 | 0.16 | 0.99 | 1.07 | 1.06 | 0.13 |
| TUT 5 | 61.1 | 0.99 | 1.17 | 1.18 | 0.13 | 0.98 | 1.24 | 1.24 | 0.13 |
| TUT 6 | 60.1 | 1.00 | 1.79 | 1.79 | 0.07 | 1.00 | 1.58 | 1.59 | 0.05 |
| TUT 7 | 63.7 | 0.99 | 1.53 | 1.54 | 0.05 | 1.00 | 1.40 | 1.43 | 0.05 |
| UEX B1 | 52.5 | 0.47 | 1.88 | 1.75 | 0 | 0.52 | 2.05 | 1.88 | 0 |
| UEX B2 | 26.3 | 0.74 | 1.73 | 1.47 | 0.05 | 0.82 | 1.66 | 1.53 | 0.05 |
| UEX B3 | 10.5 | 0.91 | 1.18 | 1.27 | 0 | 0.87 | 1.21 | 1.30 | 0 |
| UJI 1 | 27.6 | 0.99 | 1.02 | 1.04 | 0.17 | 0.98 | 1.09 | 1.09 | 0.08 |
| UJI 2 | 26.8 | 1.01 | 1.01 | 1.07 | 0.16 | 1.02 | 1.05 | 1.10 | 0.09 |
| UJIB 1 | 3.8 | - | 1.04 | 1.04 | 0.09 | - | 1.09 | 1.09 | 0.09 |
| UJIB 2 | 3.2 | - | 1.08 | 1.08 | 0.07 | - | 1.10 | 1.10 | 0.31 |
| UTS 1 | 30.3 | 1.03 | 1.07 | 1.05 | 0.18 | 1.01 | 1.09 | 1.09 | 0.11 |
| Average | 35.7 | 0.91 | 1.23 | 1.25 | 0.09 | 0.90 | 1.31 | 1.32 | 0.09 |

**Table 4.9** Numerical results for PCA+EWO32 and AE+EWO32 at $\mathcal{T} = 90$

| | | PCA+EWO32 | | | | AE+EWO32 | | | |
|---|---|---|---|---|---|---|---|---|---|
| Dataset | CR | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] |
| DSI 1 | 12.9 | - | 1.21 | 1.21 | 0.09 | - | 1.21 | 1.21 | 0.10 |
| DSI 2 | 12.9 | - | 1.23 | 1.23 | 0.11 | - | 1.28 | 1.28 | 0.17 |
| LIB 1 | 60.9 | 1.00 | 0.98 | 0.99 | 0.01 | 0.98 | 1.17 | 1.15 | 0.01 |
| LIB 2 | 46.0 | 0.99 | 1.45 | 1.43 | 0.06 | 0.99 | 1.75 | 1.73 | 0.07 |
| MAN 1 | 7.8 | - | 1.15 | 1.15 | 0.03 | - | 1.58 | 1.58 | 0.02 |
| MAN 2 | 19.6 | - | 1.22 | 1.22 | 0.02 | - | 1.39 | 1.39 | 0.07 |
| MINT 1 | 7.7 | - | 1.07 | 1.07 | 0.04 | - | 1.21 | 1.21 | 0.02 |
| SAH 1 | 54.3 | 1.27 | 1.22 | 2.09 | 0.07 | 1.32 | 1.00 | 1.73 | 0.05 |
| SIM 1 | 3.7 | - | 1.06 | 1.06 | 0.03 | - | 1.33 | 1.33 | 0.02 |
| TIE 1 | 47.7 | 0.03 | 1.50 | 2.95 | 0.10 | 0.24 | 1.73 | 1.23 | 0.06 |
| TUJI 1 | 21.7 | - | 1.17 | 1.17 | 0.16 | - | 1.27 | 1.27 | 0.05 |
| TUT 1 | 28.8 | 0.97 | 1.16 | 1.13 | 0.06 | 0.93 | 1.25 | 1.15 | 0.08 |
| TUT 2 | 38.1 | 0.91 | 1.09 | 1.03 | 0.21 | 0.88 | 1.21 | 1.22 | 0 |
| TUT 3 | 44.8 | 0.99 | 1.09 | 1.04 | 0.14 | 0.98 | 1.10 | 1.03 | 0.15 |
| TUT 4 | 42.1 | 0.99 | 1.09 | 1.06 | 0.18 | 0.99 | 1.06 | 1.05 | 0.12 |
| TUT 5 | 48.9 | 0.99 | 1.15 | 1.16 | 0.17 | 0.98 | 1.24 | 1.25 | 0.12 |
| TUT 6 | 48.0 | 1.00 | 1.73 | 1.73 | 0.06 | 1.00 | 1.49 | 1.49 | 0.04 |
| TUT 7 | 51.0 | 0.99 | 1.63 | 1.47 | 0.05 | 0.99 | 1.52 | 1.40 | 0.04 |
| UEX B1 | 42.0 | 0.49 | 1.97 | 1.70 | 0 | 0.47 | 2.29 | 2.18 | 0.04 |
| UEX B2 | 21.0 | 0.75 | 1.80 | 1.58 | 0.03 | 0.82 | 1.71 | 1.57 | 0.05 |
| UEX B3 | 8.4 | 0.93 | 1.15 | 1.24 | 0 | 0.89 | 1.15 | 1.20 | 0 |
| UJI 1 | 22.1 | 0.98 | 1.04 | 1.07 | 0.14 | 0.98 | 1.10 | 1.09 | 0.08 |
| UJI 2 | 21.4 | 1.02 | 1.01 | 1.06 | 0.17 | 1.02 | 1.04 | 1.09 | 0.09 |
| UJIB 1 | 3.1 | - | 1.04 | 1.04 | 0.15 | - | 1.09 | 1.09 | 0.07 |
| UJIB 2 | 2.6 | - | 1.07 | 1.07 | 0.20 | - | 1.13 | 1.13 | 0.12 |
| UTS 1 | 24.3 | 1.03 | 1.05 | 1.04 | 0.19 | 1.02 | 1.06 | 1.05 | 0.10 |
| Average | 28.5 | 0.90 | 1.24 | 1.31 | 0.09 | 0.91 | 1.32 | 1.31 | 0.07 |

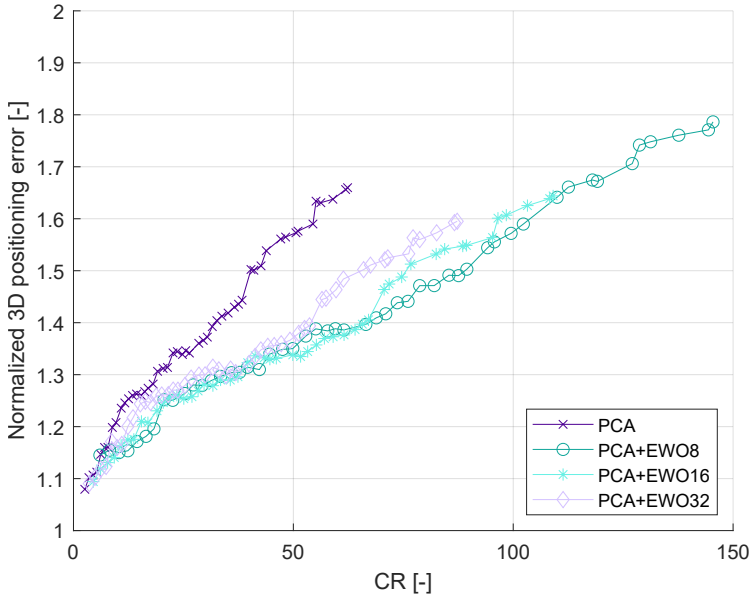


**Figure 4.3** Positioning performance and CR trade-off: combining EWOK and PCA

Similarly, the results obtained by varying the Threshold $\mathcal{T}$ in the combination of EWO$K$ with AE are depicted in Figure 4.4, showing poorer, yet still relevant performance trade-offs, compared to the implementation with PCA.

## MESSy EWO$K$

The combination of MESS and EWO$K$ compression schemes was realized on the subset of datasets that comply with the requirement of having multiple samples per training location. The implementation of EWO$K$ in terms of element-wise compression was realized separately for the set of means $\mu$ and sigmas $\sigma$, as each variable follows a different distribution. The resulting compression of EWO$K$ remains unchanged, as the relevant number of centroids (8 for EWO8) remains unchanged, while their mapping to the indexes to the centroid values is separated for $\mu$s and $\sigma$s. Technically, the CR achieved on the SDs is significantly larger, as standard deviations include real-numbered values, which for the sake of consistency is currently omitted. The provided results are normalized towards the $\alpha$ $k$-NN baseline while considering the same $k$-NN parameters. The MESS settings consider $e_{MESS} = 0.7$ and EWO$K$ initialization method is $\psi_{xtr}$.

Table 4.10 provides the numerical evaluation of the datasets compressed using

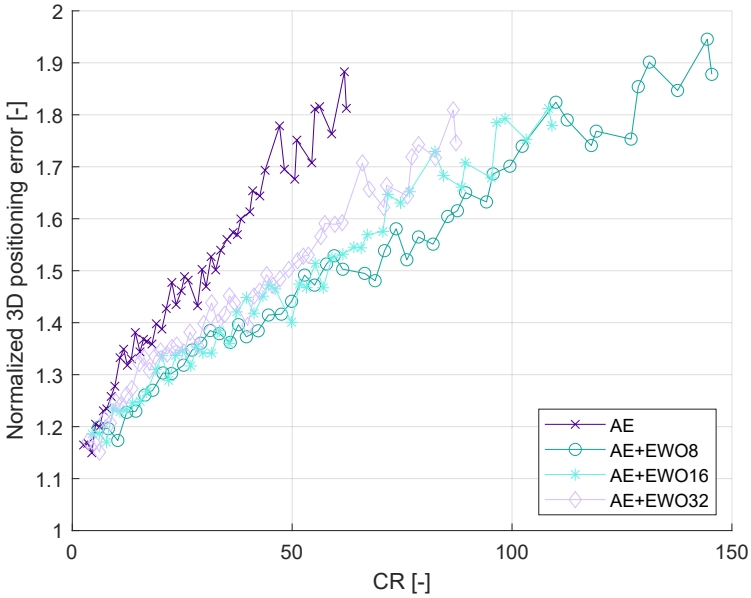


**Figure 4.4** Positioning performance and CR trade-off: combining EWOK and AE

MESS and consequent application of EWO$K$ with $K = 8$ and $\psi_{xtr}$ initialization, including the aggregated results in the bottom row of the table. The results show the considerable CRs and positioning time reduction while damaging the positioning accuracy by 13% on average. Notably, the average positioning accuracy is affected the most by the same datasets that previously degrading the accuracy when applying the standalone MESS compression in Section 3.4.

The positioning performance is only slightly improved in comparison to the previous case when considering EWO16 and EWO32, as presented in Table 4.11 and Table 4.12, respectively, whereas the achieved CRs are visibly reduced.

In terms of positioning time, the aggregated results show approx. 2-fold improvement, despite UTS 1 and UJI 2, two of the most voluminous datasets, are slowed by applying the MESS compression. The results suggest the inverse phenomenon that was observed when evaluating the cascade of models in Section 4.1, where the most voluminous datasets were reduced the most.

### Principal MESSy EWO$K$

The final set of results provided in this chapter refers to performing the radio map compression by first applying PCA, followed by sample-wise MESS, and finally

**Table 4.10** Numerical results for MESSy EWOK, $K = 8$

| Dataset | CR | $\tilde{\zeta}_{\mathcal{B}\alpha}$ [-] | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] |
|---|---|---|---|---|---|---|
| DSI 1 | 6.9 | - | - | 1.07 | 1.07 | 0.37 |
| LIB 1 | 14.0 | - | 1.00 | 0.95 | 0.96 | 0.25 |
| LIB 2 | 14.0 | - | 1.00 | 1.29 | 1.28 | 0.33 |
| MAN 1 | 128.3 | - | - | 1.14 | 1.14 | 0.01 |
| MAN 2 | 11.7 | - | - | 1.05 | 1.05 | 0.15 |
| MINT 1 | 30.7 | - | - | 1.01 | 1.01 | 0.02 |
| SIM 1 | 11.7 | - | - | 1.02 | 1.02 | 0.06 |
| TUJI 1 | 5.8 | - | - | 1.34 | 1.34 | 0.45 |
| UJI 1 | 24.8 | 1.00 | 0.85 | 1.52 | 1.46 | 0.61 |
| UJI 2 | 12.3 | 1.00 | 0.99 | 1.03 | 1.08 | 1.59 |
| UJIB 1 | 35.6 | - | - | 1.02 | 1.02 | 0.05 |
| UJIB 2 | 28.0 | - | - | 1.23 | 1.23 | 0.11 |
| UTS 1 | 7.2 | - | 0.99 | 1.06 | 1.08 | 3.04 |
| Average | 25.5 | 1.00 | 0.99 | 1.13 | 1.13 | 0.54 |

reducing the size of each element using EWO$K$. Also here each method's parameters remain consistent throughout this chapter's evaluation.

The results of Principal MESSy EWO8 are listed in Table 4.13, showing exponentially higher CRs than in the previous results, caused by the multiplicative nature of CR when combining the individual lossy compression schemes. The table shows high variation of $\tilde{\varepsilon}_{3D\alpha}$ across datasets, ranging from the improved performance of 0.96 in the dataset MINT 1 to excessive errors of 5.88 in the dataset DSI 1. Consequently, the aggregated results indicate an expected two-fold degradation of the positioning performance, while in the majority of the cases the deterioration is lower. Apart from substantial CRs, the positioning times are strongly reduced as well, especially in low-sample datasets.

Table 4.14 and Table 4.15 both show lower CRs and almost identical positioning performance of the Principal MESSy EWO16 and EWO32 schemes, compared to the implementation with $K = 8$. The variation of the performance of the individual datasets is minimal (as is in the case of stand-alone EWO$K$), while the positioning

**Table 4.11** Numerical results for MESSy EWOK, $K = 16$

| Dataset | CR | $\tilde{\zeta}_{\mathcal{B}\alpha}$ [-] | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] |
|---|---|---|---|---|---|---|
| DSI 1 | 5.2 | - | - | 1.04 | 1.04 | 0.28 |
| LIB 1 | 10.5 | - | 1.00 | 0.94 | 0.94 | 0.21 |
| LIB 2 | 10.5 | - | 1.00 | 1.27 | 1.26 | 0.25 |
| MAN 1 | 96.3 | - | - | 1.14 | 1.14 | 0.01 |
| MAN 2 | 8.8 | - | - | 1.08 | 1.08 | 0.15 |
| MINT 1 | 23.0 | - | - | 0.94 | 0.94 | 0.03 |
| SIM 1 | 8.8 | - | - | 0.94 | 0.94 | 0.06 |
| TUJI 1 | 4.4 | - | - | 1.31 | 1.31 | 0.46 |
| UJI 1 | 18.6 | 1.00 | 0.85 | 1.57 | 1.55 | 0.69 |
| UJI 2 | 9.2 | 1.00 | 0.99 | 1.01 | 1.05 | 1.68 |
| UJIB 1 | 26.7 | - | - | 1.06 | 1.06 | 0.04 |
| UJIB 2 | 21.0 | - | - | 1.23 | 1.23 | 0.09 |
| UTS 1 | 5.4 | - | 0.96 | 1.10 | 1.09 | 3.13 |
| Average | 19.1 | 1.00 | 0.98 | 1.12 | 1.12 | 0.54 |

time $\tilde{\tau}_{\alpha}$ is unaffected by the number of clusters as well.

Consequently, EWO$K$ with $K = 8$ is recommended to be applied when combined with additional compression schemes to maximize the achieved CR.

## 4.3 Concluding Remarks

The techniques introduced in this chapter target reducing the computational requirements of fingerprinting-based positioning systems while sustaining or improving their positioning capabilities.

The proposed cascade of models addresses the positioning model directly by implementing a sequence of classifiers to iteratively reduce the training dataset size for the final regression, instead of utilizing a single model that processes the whole database. The results show that the positioning time is severely reduced across the deployments, especially the ones consisting of voluminous training sets while boosting the positioning performance at the same time. The best-performing models were

**Table 4.12** Numerical results for MESSy EWOK, $K = 32$

| Dataset | CR | $\tilde{\zeta}_{\mathcal{B}\alpha}$ [-] | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] |
|---|---|---|---|---|---|---|
| DSI 1 | 4.2 | - | - | 1.02 | 1.02 | 0.37 |
| LIB 1 | 8.4 | - | 1.00 | 0.95 | 0.95 | 0.24 |
| LIB 2 | 8.4 | - | 0.99 | 1.27 | 1.26 | 0.28 |
| MAN 1 | 77.0 | - | - | 1.09 | 1.09 | 0.01 |
| MAN 2 | 7.0 | - | - | 0.98 | 0.98 | 0.25 |
| MINT 1 | 18.4 | - | - | 0.90 | 0.90 | 0.02 |
| SIM 1 | 7.0 | - | - | 0.92 | 0.92 | 0.06 |
| TUJI 1 | 3.5 | - | - | 1.26 | 1.26 | 0.46 |
| UJI 1 | 14.9 | 1.00 | 0.86 | 1.53 | 1.51 | 0.77 |
| UJI 2 | 7.4 | 1.00 | 1.00 | 1.00 | 1.04 | 1.63 |
| UJIB 1 | 21.4 | - | - | 1.12 | 1.12 | 0.08 |
| UJIB 2 | 16.8 | - | - | 1.21 | 1.21 | 0.05 |
| UTS 1 | 4.3 | - | 0.98 | 1.10 | 1.10 | 3.28 |
| Average | 15.3 | 1.00 | 0.99 | 1.10 | 1.10 | 0.58 |

selected based on the prior validation phase, which was evaluated on a part of the training samples. In the majority of cases, this led to improved positioning performance on the testing set in terms of accuracy and greatly reduced positioning times. Nevertheless, in case there is a mismatch between the training and testing (validation) distributions of samples, the final positioning performance can underperform (e.g. the floor hit performance of TIE 1 in Table 4.5). Overall results suggest that utilizing the cascade of ML models for building and floor estimation, followed by a $k$-NN positioning estimator guarantees to radically improve the positioning time and consequently reduce the positioning effort of the device while enabling possible gains in terms of positioning accuracy as well.

The chapter further evaluates the achieved performance of combining the individual radio map compression schemes proposed in Chapter 3 in terms of the considered metrics. The combination of methods enables achieving higher CRs and significantly reduces the positioning time, while having limited impact on the resulting positioning performance, in the majority of cases. The results show that an excessive combi-

**Table 4.13** Numerical results for Principal MESSy EWOK, $K = 8$

| Dataset | CR | $\tilde{\zeta}_{\mathcal{B}\alpha}$ [-] | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] |
|---|---|---|---|---|---|---|
| DSI 1 | 30.3 | - | - | 5.88 | 5.88 | 0.34 |
| LIB 1 | 270.7 | - | 0.97 | 1.27 | 1.21 | 0.09 |
| LIB 2 | 183.9 | - | 0.94 | 1.76 | 1.64 | 0.13 |
| MAN 1 | 399.3 | - | - | 2.40 | 2.40 | 0.01 |
| MAN 2 | 163.3 | - | - | 2.71 | 2.71 | 0.06 |
| MINT 1 | 67.5 | - | - | 0.96 | 0.96 | 0.02 |
| SIM 1 | 18.7 | - | - | 1.01 | 1.01 | 0.05 |
| TUJI 1 | 50.2 | - | - | 1.87 | 1.87 | 0.38 |
| UJI 1 | 201.8 | 1.00 | 0.77 | 1.79 | 1.61 | 0.12 |
| UJI 2 | 95.2 | 1.00 | 0.80 | 1.36 | 1.38 | 0.26 |
| UJIB 1 | 53.4 | - | - | 1.34 | 1.34 | 0.03 |
| UJIB 2 | 36.2 | - | - | 1.62 | 1.62 | 0.03 |
| UTS 1 | 51.4 | - | 0.93 | 1.35 | 1.23 | 0.44 |
| Average | 124.8 | 1.00 | 0.95 | 1.95 | 1.91 | 0.15 |

nation of the compression methods can lead to severe performance deterioration, as is the case of Principal MESSy EWO8 of DSI 1 in Table 4.13, which deteriorates the positioning performance 5.88 times. Yet, the same table shows the potential for improved performance with the same compression scheme. In the scenario of MINT 1 dataset, the 67.5-fold compression led to improved positioning performance by 4% and reduced positioning time by 98%. The results indicate that the effectiveness of a lossy compression method is often data-specific and needs to be evaluated before being implemented in any positioning system.

To summarize the overall performance of all proposed radio map compression schemes and their combinations, Table 4.16 provides the comparative performance of all methods at the common parameter settings. The number of clusters of EWO$K$ is set as $K = 8$ with initialization method $\psi_{xtr}$, the exponent parameter of MESS is set as $e_{MESS} = 0.7$, and the Threshold of PCA $\mathcal{T} = 90$. The hidden dimension of the AE corresponds to the number of considered components of PCA. The table was obtained by aggregating the results on the datasets applicable across all

**Table 4.14** Numerical results for Principal MESSy EWOK, $K = 16$

| Dataset | CR | $\tilde{\zeta}_{\mathcal{B}\alpha}$ [-] | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] |
|---|---|---|---|---|---|---|
| DSI 1 | 22.7 | - | - | 5.36 | 5.36 | 0.32 |
| LIB 1 | 203.0 | - | 0.98 | 1.21 | 1.18 | 0.08 |
| LIB 2 | 137.9 | - | 0.94 | 1.72 | 1.60 | 0.15 |
| MAN 1 | 299.4 | - | - | 2.05 | 2.05 | 0.01 |
| MAN 2 | 122.5 | - | - | 2.46 | 2.46 | 0.07 |
| MINT 1 | 50.6 | - | - | 1.03 | 1.03 | 0.02 |
| SIM 1 | 14.0 | - | - | 0.96 | 0.96 | 0.05 |
| TUJI 1 | 37.7 | - | - | 1.85 | 1.85 | 0.41 |
| UJI 1 | 151.4 | 1.00 | 0.77 | 1.74 | 1.62 | 0.12 |
| UJI 2 | 71.4 | 1.00 | 0.81 | 1.33 | 1.34 | 0.27 |
| UJIB 1 | 40.0 | - | - | 1.31 | 1.31 | 0.03 |
| UJIB 2 | 27.2 | - | - | 1.62 | 1.62 | 0.03 |
| UTS 1 | 38.6 | - | 0.92 | 1.36 | 1.27 | 0.46 |
| Average | 93.6 | 1.00 | 0.95 | 1.85 | 1.82 | 0.16 |

scenarios (not considering the ones with a single measurement per location) and by normalizing all individual results towards baseline $\alpha$. The $k$-NN setting used for the actual positioning estimation was $k = 1$, Sørensen distance, and a positive data representation. The full table is available in Appendix B, Table B.1.

By far, the highest CR was obtained by combining all considered methods, while stand-alone EWO8 reduced the dataset by only 2.3-fold (by a ratio of 7/3). In terms of positioning error, EWO8 and MESS outperformed the other methods, while Principal MESSy EWO8 deteriorated the performance the most (considering the underperforming DSI 1 dataset). AE and AE+EWO8 achieved the largest positioning time reduction of 92%.

The numbers provided in Table 4.16 can be interpreted in the following way, considering the MESSy EWO8 as an example. The average CR of 25.5 indicates that the compressed radio map takes 3.9% of the original radio map's size, and consequently only the 3.9% of the original data volume needs to be processed by the positioning model, leading to solid reduction of the positioning time. The position-

**Table 4.15** Numerical results for Principal MESSy EWOK, $K = 32$

| Dataset | CR | $\tilde{\zeta}_{\mathcal{B}\alpha}$ [-] | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] |
|---|---|---|---|---|---|---|
| DSI 1 | 18.2 | - | - | 5.39 | 5.39 | 0.49 |
| LIB 1 | 162.4 | - | 0.98 | 1.20 | 1.16 | 0.10 |
| LIB 2 | 110.3 | - | 0.93 | 1.80 | 1.65 | 0.12 |
| MAN 1 | 239.6 | - | - | 2.04 | 2.04 | 0.01 |
| MAN 2 | 98.0 | - | - | 2.26 | 2.26 | 0.07 |
| MINT 1 | 40.5 | - | - | 1.04 | 1.04 | 0.02 |
| SIM 1 | 11.2 | - | - | 0.93 | 0.93 | 0.05 |
| TUJI 1 | 30.1 | - | - | 1.79 | 1.79 | 0.40 |
| UJI 1 | 121.1 | 1.00 | 0.77 | 1.69 | 1.57 | 0.13 |
| UJI 2 | 57.1 | 1.00 | 0.80 | 1.36 | 1.37 | 0.27 |
| UJIB 1 | 32.0 | - | - | 1.28 | 1.28 | 0.04 |
| UJIB 2 | 21.7 | - | - | 1.64 | 1.64 | 0.05 |
| UTS 1 | 30.8 | - | 0.93 | 1.32 | 1.20 | 0.46 |
| Average | 74.9 | 1.00 | 0.95 | 1.83 | 1.79 | 0.17 |

**Table 4.16** Averaged numerical results for all considered compression schemes

| Method | CR | $\tilde{\zeta}_{\mathcal{B}\alpha}$ [-] | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] |
|---|---|---|---|---|---|---|
| EWO8 | 2.3 | 1.00 | 1.00 | 1.03 | 1.03 | 1.00 |
| PCA | 13.9 | 1.00 | 1.00 | 1.12 | 1.12 | 0.09 |
| AE | 13.9 | 1.00 | 1.00 | 1.28 | 1.28 | 0.08 |
| MESS | 10.9 | 1.00 | 0.96 | 1.06 | 1.06 | 0.73 |
| PCA+EWO8 | 32.5 | 1.00 | 1.00 | 1.14 | 1.15 | 0.10 |
| AE+EWO8 | 32.5 | 1.00 | 1.00 | 1.25 | 1.25 | 0.08 |
| MESSy EWO8 | 25.5 | 1.00 | 0.97 | 1.13 | 1.13 | 0.54 |
| Principal MESSy EWO8 | 124.8 | 1.00 | 0.88 | 1.95 | 1.91 | 0.15 |

ing performance deterioration of 13% means that if the original positioning system achieved 3 m of mean positioning error before compression (as is the case in many of the considered datasets), the positioning error after applying the compression scheme increases by 40 cm. The relevance of the provided trade-off is evident.

The relevant observation regarding the positioning time reduction, based on the results provided in this chapter, indicates that in the case of voluminous datasets, the cascade model reduces the computational requirements by the highest amount, while in case the datasets contain a lower amount of samples, the positioning time is still reduced, but to a lesser extent. On contrary, the proposed MESS compression accelerates the positioning the strongest in case small-scale datasets are considered. The feature-wise PCA and AE compression schemes reduce the positioning time without a direct link to the dataset volume.

Overall, the methods proposed within this and the previous chapter are capable of reducing the radio map, accelerating the positioning process, as well as in many cases boosting, or sustaining the positioning accuracy of the dataset. The utilization of the proposed methods may strongly improve the characteristics of the positioning system, while the utilization of the individual methods needs to be validated before implementing in a real-world application. This may be achieved by performing a similar validation phase as was described in Section 4.1 while testing for the compression performance.

## 4.4 Author's Contributions

The main Author's contributions to the field of fingerprinting-based positioning time optimization discussed in this chapter may be summarized in the following terms:

- Introducing the relevant ML models that can be utilized for efficient localization in indoor positioning systems.
- Experimentally evaluating the best-performing models for the building, floor, and position estimation tasks, showing that in terms of positioning accuracy *k*-NN model is globally unmatched.
- Proposing a novel cascading scheme to boost positioning performance. Based on the prior validation, proposing an optimal cascade of positioning models to universally fit each considered positioning scenario.

- Designing a deployment-specific cascade of models for each dataset.
- Numerically evaluating the proposed solutions on the test sets, achieving 80% lower positioning times on average, with 98% reduction on the voluminous ones, compared to stand-alone $k$-NN while achieving a slight improvement in average positioning performance.
- Designing a scheme for implementing a multi-dimensional radio map compression within an indoor positioning system.
- Evaluating the following combinations of the radio map compression methods in terms of CR, positioning performance, and positioning time:
  - Feature-wise EWO$K$ (Feature-wise PCA or AE and bit-level EWO$K$)
  - MESSy EWO$K$ (Sample-wise MESS and bit-level EWO$K$)
  - Principal MESSy EWO$K$ (incorporating PCA, MESS, and EWO$K$ into a single scheme)
- Listing the comparative summary of the positioning and compression performances of the individual methods and their combinations.
- Discussing the advantages, challenges, and limitations of combining the individual compression schemes.

CHAPTER 5

# THESIS SUMMARY AND OPEN RESEARCH DIRECTIONS

The goal of this thesis has been to contribute to the current SotA by proposing novel methods for data compression, that will lead to the increased energy efficiency of the nowadays' wearable-based systems, alleviating the storage, data transfer, and gathering requirements, as well as boosting their positioning capabilities. The elementary requirements of lossy compression techniques are the capability of reducing the volume without damaging the meaningful information within the data, the energy efficiency of the compression algorithm, and the latency caused by the compression. The additional requirement on the lossy compressions applied within the positioning system is preserving the positioning capabilities of the data after the compression was applied. In terms of the time-series sensor-based data, the required information within the data that has to be preserved depends on the specific application.

In Chapter 1 four research questions were identified. The answers to these questions were intensively discussed in the previous chapters and their concise answers may be summarized as follows:

RQ1. *Which lossy compression mechanisms can be implemented for energy-efficient, delay-sensitive wearable data gathering, transfer, and storage?*

In light of the discussion in Chapter 2, numerous compression methods may find their relevant application in terms of wearable sensor-based data gathering, transfer, and storage. Nevertheless, in terms of considered time-series data, the performed evaluation clearly indicates that across the considered trade-offs and performance metrics, the proposed DLTC method outperforms the other considered compression techniques. In case of the emphasis on minimized system latency, the proposed ALTC technique should be considered due to its delay-free implementation.

RQ2. *To what extent can the bit-level, feature-wise, and sample-wise reduction of the radio map support accurate positioning while saving resources in data storage and transfer?*

The proposed radio map compression techniques discussed in Chapter 3 were evaluated in terms of their achievable CR and their impact on positioning performance. The implementation of the novel bit-level compression EWO*K* enables preserving resources in terms of storage and transfer while generally sustaining the positioning performance. The feature-wise compression schemes enable manyfold dimensionality reduction, while additionally accelerating the positioning speed in the process while slightly deteriorating the positioning accuracy. The proposed sample-wise compression scheme denoted MESS has the potential to improve the positioning accuracy, while at the same time reducing the dataset volume and the time required for positioning.

RQ3. *How to compensate for k-NN's drawback of computationally expensive prediction on voluminous datasets?*

The drawback of *k*-NN's lengthy compression has been addressed directly on the positioning model by implementing the novel cascade system introduced in Chapter 4, which splits the training dataset into smaller parts, which are then processed by the *k*-NN with significantly reduced effort. At the same time, the model proposes alternative methods to be used for building and floor classification, without influencing the final positioning performance, while proving that the *k*-NN's performance in terms 2D accuracy is unmatched. The cascade reduced the prediction time by up to 98% while boosting the positioning

performance in terms of accuracy by up to 38% in terms of positioning error. Alternatively, reducing the radio map also reduces the amount of data that the $k$-NN has to process, resulting in relevant savings in terms of computational resources and processing time. The reduction strongly depends on the utilized radio map compression scheme, as well as on the training data, while having the potential to accelerate the positioning speed up to 100-fold, according to the results provided in Chapter 4. The results show that the cascade of models boosts the positioning time, especially of the voluminous datasets, while the MESS compression scheme shows higher efficiency in terms of positioning time reduction when small-scale datasets are considered.

RQ4. *How to implement a multi-dimensional compression of the radio map to boost the performance efficiency of the positioning system?*

The results provided in Chapter 4 show that by combining the individual radio map compression methods, the achieved compression ratio drastically increases by a factor of (up to) hundreds, compared to the scenario of utilizing a single compression scheme. The combination of methods generally boosts the performance in terms of positioning speed (by 90% on average for PCA+EWO8) at the cost of slightly increased positioning errors (14% on average). The work concludes that multi-dimensional compression has the potential to significantly improve the overall positioning performance if the compression schemes are carefully selected and parameterized.

## Future Work

The field of wearable-based computing is a relatively young branch of science, which still requires plenty of innovative ideas to be fully explored and exploited. There are numerous unanswered questions in the field of wearables, localization, and sensing approaches, where compression techniques may be implemented to boost energy efficiency in a less traditional way.

Small-scale wearables, such as patches or implants, which operate mostly by collecting sensor-based data are perfect candidates to operate with an efficient lossy compression scheme. Their applications are still aimed on eHealth and their availability is limited, but the advantages these small monitoring devices offer are undisputed.

The progress towards wireless VR and Augmented Reality (AR) immersion is

imminent, and the lossy compression mechanism reducing the volumes of the data these technologies require provides possibilities that may accelerate their development.

Implementing efficient, large-scale fingerprinting positioning systems is nowadays challenging as the volume of the fingerprinting database naturally increases with the increasing area of deployment. The results provided in this thesis will hopefully lay the basis for realizing the idea of the large-scale positioning system, applicable to any device without the excessive requirements for storage or computational resources.

As a part of the Author's future work, publishing and extending the novel methods provided within this thesis is currently considered, including a paper extending the idea of MESS compression scheme, publishing the TUJI 1 dataset as open-source, and elaborating on the idea of specifying the compression latency using the delay as a universal metric. Additionally, some of the future directions considering or built on the schemes proposed in this thesis that further contribute to SotA include utilizing the DLTC in the scope of communications and positioning as a compression mechanism for reducing the radio-based data along the user path or extending the DLTC method into a multi-dimensional compression scheme.

# REFERENCES


[1] D. Le Blanc, Towards integration at last? The sustainable development goals as a network of targets, *Sustainable Development*, vol. 23, no. 3, 176–187, 2015.

[2] 3GPP, "Technical Specification Group Radio Access Network; Study on further enhancements to LTE Device to Device (D2D), User Equipment (UE) to network relays for Internet of Things (IoT) and wearables; (Release 15)," 3GPP, 36.746, Apr. 2018, Version 15.1.1.

[3] K. Lauber, G. Wort, F. Gillison, D. Thompson, D. Augustine and O. Peacock, Patient views of the role of wearable technology to enhance cardiac rehabilitation, in *United Kingdom Society for Behavioural Medicine annual conference*, 2020.

[4] S. Iqbal, I. Mahgoub, E. Du, M. A. Leavitt and W. Asghar, Advances in healthcare wearable devices, *NPJ Flexible Electronics*, vol. 5, no. 1, 1–14, 2021.

[5] Q. Lyu, S. Gong, J. Yin, J. M. Dyson and W. Cheng, Soft wearable healthcare materials and devices, *Advanced healthcare materials*, vol. 10, no. 17, 2100577, 2021.

[6] M.-Y. Jeng, T.-M. Yeh and F.-Y. Pai, A Performance Evaluation Matrix for Measuring the Life Satisfaction of Older Adults Using eHealth Wearables, in *Healthcare*, MDPI, vol. 10, 2022, 605.

[7] E. Svertoka, S. Saafi, A. Rusu-Casandra, R. Burget, I. Marghescu, J. Hosek and A. Ometov, Wearables for industrial work safety: A survey, *Sensors*, vol. 21, no. 11, 3844, 2021.

[8] Z. Song, Z. Cao, Z. Li, J. Wang and Y. Liu, Inertial motion tracking on mobile and wearable devices: recent advancements and challenges, *Tsinghua Science and Technology*, vol. 26, no. 5, 692–705, 2021.

[9] S. Dhanekar and K. Rangra, Wearable dosimeters for medical and defence applications: A state of the art review, *Advanced Materials Technologies*, vol. 6, no. 5, 2000895, 2021.

[10] A. Ometov, V. Shubina, L. Klus, J. Skibińska, S. Saafi, P. Pascacio, L. Flueratoru, D. Q. Gaibor, N. Chukhno, O. Chukhno *et al.*, A survey on wearable technology: History, state-of-the-art and current challenges, *Computer Networks*, vol. 193, 108074, 2021.

[11] L. Klus, E. S. Lohan, C. Granell and J. Nurmi, Crowdsourcing Solutions for Data Gathering from Wearables, in Proc. of XXXV Finnish URSI Convention on Radio Science, 2019.

[12] L. Klus, E.-S. Lohan, C. Granell and J. Nurmi, Lossy compression methods for performance-restricted wearable devices, in *WiP Proceedings of the International Conference on Localization and GNSS (ICL-GNSS 2020)*, CEUR-WS, 2020.

[13] L. Klus, R. Klus, E. S. Lohan, C. Granell, J. Talvitie, M. Valkama and J. Nurmi, Direct lightweight temporal compression for wearable sensor data, *IEEE Sensors Letters*, vol. 5, no. 2, 1–4, 2021.

[14] L. Klus, D. Quezada-Gaibor, J. Torres-Sospedra, E. S. Lohan, C. Granell and J. Nurmi, RSS fingerprinting dataset size reduction using feature-wise adaptive k-means clustering, in *2020 12th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, IEEE, 2020, 195–200.

[15] D. Quezada-Gaibor, L. Klus, J. Torres-Sospedra, E. S. Lohan, J. Nurmi and J. Huerta, Improving DBSCAN for indoor positioning using Wi-Fi radio maps in wearable and IoT devices, in *2020 12th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, IEEE, 2020, 208–213.

[16] R. Klus, L. Klus, J. Talvitie, J. Pihlajasalo, J. Torres-Sospedra and M. Valkama, Transfer learning for convolutional indoor positioning systems, in *2021 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, IEEE, 2021, 1–8.

[17] L. Klus, D. Quezada-Gaibor, J. Torres-Sospedra, E. S. Lohan, C. Granell and J. Nurmi, Towards Accelerated Localization Performance Across Indoor Positioning Datasets, in *2022 International Conference on Localization and GNSS (ICL-GNSS)*, IEEE, 2022, 1–7.

[18] D. Quezada-Gaibor, L. Klus, J. Torres-Sospedra, E. S. Lohan, J. Nurmi, C. Granell and J. Huerta, Data Cleansing for Indoor Positioning Wi-Fi Fingerprinting Datasets, 1–6, 2022.

[19] R. Klus, L. Klus, D. Solomitckii, M. Valkama and J. Talvitie, Deep learning based localization and HO optimization in 5G NR networks, in *2020 International Conference on Localization and GNSS (ICL-GNSS)*, IEEE, 2020, 1–6.

[20] R. Klus, L. Klus, D. Solomitckii, J. Talvitie and M. Valkama, Deep learning-based cell-level and beam-level mobility management system, *Sensors*, vol. 20, no. 24, 7124, 2020.

[21] J. Torres-Sospedra, I. Silva, L. Klus, D. Quezada-Gaibor, A. Crivello, P. Barsocchi, C. Pendão, E. S. Lohan, J. Nurmi and A. Moreira, Towards ubiquitous indoor positioning: Comparing systems across heterogeneous datasets, in *2021 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, IEEE, 2021, 1–8.

[22] E. Park, User acceptance of smart wearable devices: An expectation-confirmation model approach, *Telematics and Informatics*, vol. 47, 101318, 2020.

[23] M. D. Peláez-Coca, A. Hernando, J. Lázaro and E. Gil, Impact of the PPG sampling rate in the pulse rate variability indices evaluating several fiducial points in different pulse waveforms, *IEEE Journal of Biomedical and Health Informatics*, vol. 26, no. 2, 539–549, 2021.

[24] E. Nemati, M. J. Deen and T. Mondal, A wireless wearable ECG sensor for long-term applications, *IEEE Communications Magazine*, vol. 50, no. 1, 36–43, 2012.

[25] F. A. Kraemer, A. E. Braten, N. Tamkittikhun and D. Palma, Fog computing in healthcare–a review and discussion, *IEEE Access*, vol. 5, 9206–9222, 2017.

[26] A. Mosenia, S. Sur-Kolay, A. Raghunathan and N. K. Jha, Wearable medical sensor-based system design: A survey, *IEEE Transactions on Multi-Scale Computing Systems*, vol. 3, no. 2, 124–138, 2017.

[27] H. Banaee, M. U. Ahmed and A. Loutfi, Data mining for wearable sensors in health monitoring systems: a review of recent trends and challenges, *Sensors*, vol. 13, no. 12, 17472–17500, 2013.

[28] K. Sayood, *Introduction to data compression*. Morgan Kaufmann, 2017.

[29] S. Faye, N. Louveton, S. Jafarnejad, R. Kryvchenko and T. Engel, An Open Dataset for Human Activity Analysis using Smart Devices, 2017.

[30] R. Furberg, J. Brinton, M. Keating and A. Ortiz, *Crowd-sourced Fitbit datasets 03.12.2016-05.12.2016*, Zenodo, May 2016. DOI: 10.5281/zenodo.53894. [Online]. Available: https://doi.org/10.5281/zenodo.53894.

[31] S. Rani, H. Babbar, S. Coleman, A. Singh and H. M. Aljahdali, An efficient and lightweight deep learning model for human activity recognition using smartphones, *Sensors*, vol. 21, no. 11, 3845, 2021.

[32] H. Alrehamy and C. Walker, (Personal Data Lake) SemLinker: automating big data integration for casual users, *Journal of Big Data*, vol. 5, Mar. 2018. DOI: 10.1186/s40537-018-0123-x.

[33] F. Niemann, C. Reining, F. Moya Rueda, N. R. Nair, J. A. Steffens, G. A. Fink and M. Ten Hompel, Lara: Creating a dataset for human activity recognition in logistics using semantic attributes, *Sensors*, vol. 20, no. 15, 4083, 2020.

[34] M. Kachuee, S. Fazeli and M. Sarrafzadeh, ECG heartbeat classification: A deep transferable representation, in *2018 IEEE international conference on healthcare informatics (ICHI)*, IEEE, 2018, 443–444.

[35] *ECG Heartbeat Categorization Dataset*, `https://www.kaggle.com/datasets/shayanfazeli/heartbeat?resource=download&page=1`, Accessed: 2022-11-06.

[36] J. Guckert, The use of FFT and MDCT in MP3 audio compression, *Math 2270 Tutorial*, 13, 2012.

[37] K. R. Rao and J. J. Hwang, *Techniques and standards for image, video, and audio coding*. Prentice-Hall, Inc., 1996.

[38] P. Kavitha, A survey on lossless and lossy data compression methods, *International Journal of Computer Science & Engineering Technology*, vol. 7, no. 03, 110–114, 2016.

[39] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, *Introduction to algorithms*. MIT press, 2022.

[40] J. Lin, E. Keogh, S. Lonardi and B. Chiu, 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, 2–11.

[41] E. P. Capo-Chichi, H. Guyennet and J.-M. Friedt, K-RLE: a new data compression algorithm for wireless sensor network, in *2009 Third International Conference on Sensor Technologies and Applications*, IEEE, 2009, 502–507.

[42] E. Keogh, K. Chakrabarti, M. Pazzani and S. Mehrotra, Locally adaptive dimensionality reduction for indexing large time series databases, in *Proc. of the 2001 ACM SIGMOD international conference on Management of data*, 2001.

[43] T. Schoellhammer, B. Greenstein, E. Osterweil, M. Wimbrow and D. Estrin, Lightweight temporal compression of microclimate datasets, 2004.

[44] T. Altstidl, S. Kram, O. Herrmann, M. Stahlke, T. Feigl and C. Mutschler, Accuracy-aware compression of channel impulse responses using deep learning, in *2021 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, IEEE, 2021, 1–8.

[45] D. Del Testa and M. Rossi, Lightweight lossy compression of biometric patterns via denoising autoencoders, *IEEE Signal Processing Letters*, vol. 22, no. 12, 2304–2308, 2015.

[46] M. Hooshmand, D. Zordan, D. Del Testa, E. Grisan and M. Rossi, Boosting the battery life of wearables for health monitoring through the compression of biosignals, *IEEE Internet of Things Journal*, vol. 4, no. 5, 1647–1662, 2017.

[47] S. Vadrevu and M. S. Manikandan, A New Quality-Aware Quality-Control Data Compression Framework for Power Reduction in IoT and Smartphone PPG Monitoring Devices, *IEEE Sensors Letters*, vol. 3, no. 7, 1–4, 2019.

[48] G. Giorgi, A combined approach for real-time data compression in wireless body sensor networks, *IEEE Sensors Journal*, vol. 17, 6129–6135, 2017.

[49] N. Ahmed, T. Natarajan and K. R. Rao, Discrete cosine transform, *IEEE transactions on Computers*, vol. 100, no. 1, 90–93, 1974.

[50] G. Strang, The discrete cosine transform, *SIAM review*, vol. 41, no. 1, 135–147, 1999.

[51] J. Talvitie, M. Renfors and E. S. Lohan, Novel indoor positioning mechanism via spectral compression, *IEEE Communications Letters*, vol. 20, no. 2, 352–355, 2015.

[52] J. Talvitie, M. Renfors, M. Valkama and E. S. Lohan, Method and analysis of spectrally compressed radio images for mobile-centric indoor localization, *IEEE Transactions on Mobile Computing*, vol. 17, no. 4, 845–858, 2017.

[53] B. Rajesh, M. Javed, S. Srivastava *et al.*, DCT-COMPCNN: A novel image classification network using JPEG compressed dct coefficients, in *2019 IEEE Conference on Information and Communication Technology*, IEEE, 2019, 1–6.

[54] A. Hammouch *et al.*, Handwritten digit recognition based on DCT features and SVM classifier, in *2014 Second world conference on complex systems (WCCS)*, IEEE, 2014, 13–16.

[55] S. Nakajima, H. Iwaki, M. Ikebuchi, R. Kato and T. Toriu, Wearable Accelerometer for Numerical Diagnosis of Human Walk Using DCT, in *2008 Digest of Technical Papers-International Conference on Consumer Electronics*, IEEE, 2008, 1–2.

[56] J. Lin, E. Keogh, L. Wei and S. Lonardi, Experiencing SAX: a novel symbolic representation of time series, *Data Mining and knowledge discovery*, vol. 15, no. 2, 107–144, 2007.

[57] J. Shieh and E. Keogh, i SAX: indexing and mining terabyte sized time series, in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2008, 623–631.

[58] Y. Mohammad and T. Nishida, Robust learning from demonstrations using multidimensional SAX, in *2014 14th International Conference on Control, Automation and Systems (ICCAS 2014)*, IEEE, 2014, 64–71.

[59] Y. Yu, Y. Zhu, D. Wan, H. Liu and Q. Zhao, A novel symbolic aggregate approximation for time series, in *International Conference on Ubiquitous Information Management and Communication*, Springer, 2019, 805–822.

[60] L. Pappa, P. Karvelis, G. Georgoulas and C. Stylios, Slopewise Aggregate Approximation SAX: keeping the trend of a time series, in *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*, IEEE, 2021, 01–08.

[61] H. Tayebi, S. Krishnaswamy, A. B. Waluyo, A. Sinha and M. M. Gaber, Ra-sax: resource-aware symbolic aggregate approximation for mobile ECG analysis, in *2011 IEEE 12th international conference on mobile data management*, IEEE, vol. 1, 2011, 289–290.

[62] L. Wang, F. Lu, M. Cui and Y. Bao, Survey of methods for time series symbolic aggregate approximation, in *International Conference of Pioneering Computer Scientists, Engineers and Educators*, Springer, 2019, 645–657.

[63] Y. Liang, Efficient temporal compression in wireless sensor networks, in *2011 IEEE 36th Conference on Local Computer Networks*, IEEE, 2011, 466–474.

[64] M. J. Rubin, M. B. Wakin and T. Camp, Lossy compression for wireless seismic data acquisition, *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 9, no. 1, 236–252, 2015.

[65] R. Sharma, A data compression application for wireless sensor networks using LTC algorithm, in *IEEE International Conference on Electro/Information Technology*, 2015.

[66] V. Bashlovkina, M. Abdelaal and O. Theel, Fuzzycat: a lightweight adaptive transform for sensor data compression, in *2015 IEEE International Conference on Communication Workshop (ICCW)*, IEEE, 2015, 2756–2762.

[67] O. Sarbishei, Refined Lightweight Temporal Compression for Energy-Efficient Sensor Data Streaming, in *2019 IEEE 5th World Forum on Internet of Things*, IEEE, 2019.

[68] J. Azar, A. Makhoul, R. Darazi, J. Demerjian and R. Couturier, On the performance of resource-aware compression techniques for vital signs data in wireless body sensor networks, in *2018 IEEE Middle East and North Africa Communications Conference (MENACOMM)*, IEEE, 2018, 1–6.

[69] B. Li, O. Sarbishei, H. Nourani and T. Glatard, A multi-dimensional extension of the Lightweight Temporal Compression method, in *2018 IEEE International Conference on Big Data (Big Data)*, IEEE, 2018, 2918–2923.

[70] S. L. Halson, M. W. Bridge, R. Meeusen, B. Busschaert, M. Gleeson, D. A. Jones and A. E. Jeukendrup, Time course of performance changes and fatigue markers during intensified training in trained cyclists, *Journal of applied physiology*, vol. 93, no. 3, 947–956, 2002.

[71] R. Klus, J. Talvitie and M. Valkama, Neural network fingerprinting and GNSS data fusion for improved localization in 5G, in *2021 International Conference on Localization and GNSS (ICL-GNSS)*, IEEE, 2021, 1–6.

[72] K. Nagai, T. Fasoro, M. Spenko, R. Henderson and B. Pervan, Evaluating GNSS navigation availability in 3-D mapped urban environments, in *2020 IEEE/ION Position, Location and Navigation Symposium (PLANS)*, IEEE, 2020, 639–646.

[73] S. Subedi and J.-Y. Pyun, A survey of smartphone-based indoor positioning system using RF-based wireless technologies, *Sensors*, vol. 20, no. 24, 7230, 2020.

[74] D. Plets, W. Deprez, J. Trogh, L. Martens and W. Joseph, Joint received signal strength, angle-of-arrival, and time-of-flight positioning, in *2019 13th European Conference on Antennas and Propagation (EuCAP)*, IEEE, 2019, 1–5.

[75] 3GPP, "NG Radio Access Network (NG-RAN); Stage 2 functional specification of User Equipment (UE) positioning in NG-RAN," 3GPP, 38.305, Sep. 2022, Version 16.8.0.

[76] 3GPP, "Stage 2 functional specification of User Equipment (UE) positioning in NG-RAN," 3GPP, 38.305, Dec. 2021, Version 16.7.0.

[77] R. Klus, J. Talvitie, J. Vinogradova, J. Torsner and M. Valkama, Machine Learning Based NLOS Radio Positioning in Beamforming Networks, in *2022 IEEE 23rd International Workshop on Signal Processing Advances in Wireless Communication (SPAWC)*, IEEE, 2022, 1–5.

[78] M. Stahlke, S. Kram, F. Ott, T. Feigl and C. Mutschler, Estimating toa reliability with variational autoencoders, *IEEE Sensors Journal*, vol. 22, no. 6, 5133–5140, 2021.

[79] J. Torres-Sospedra, P. Richter, A. Moreira, G. Mendoza-Silva, E.-S. Lohan, S. Trilles, M. Matey-Sanz and J. Huerta, A comprehensive and reproducible comparison of clustering and optimization rules in Wi-Fi fingerprinting, *IEEE Trans. on Mobile Computing*, 2020.

[80] H.-A. Pham, T.-V. Le *et al.*, An Improved Weighted K-Nearest Neighbors Algorithm for High Accuracy in Indoor Localization, in *2019 25th Asia-Pacific Conference on Communications (APCC)*, IEEE, 2019, 24–27.

[81] M. Abid, P. Compagnon and G. Lefebvre, Improved CNN-based Magnetic Indoor Positioning System using Attention Mechanism, in *2021 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, IEEE, 2021, 1–8.

[82] J. Talvitie, E. S. Lohan and M. Renfors, The effect of coverage gaps and measurement inaccuracies in fingerprinting based indoor localization, in *International Conference on Localization and GNSS 2014 (ICL-GNSS 2014)*, 2014, 1–6. DOI: 10.1109/ICL-GNSS.2014.6934181.

[83] S. Parrino, G. Peruzzi and A. Pozzebon, LoPATraN: Low Power Asset Tracking by Means of Narrow Band IoT (NB-IoT) Technology, *Sensors*, vol. 21, no. 11, 3772, 2021.

[84] C. K. M. Lee, C. Ip, T. Park and S. Chung, A bluetooth location-based indoor positioning system for asset tracking in warehouse, in *2019 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, IEEE, 2019, 1408–1412.

[85] W. Liu, Improvement of navigation of Mobile Robotics based on IoT System, in *2021 IEEE International Conference on Robotics, Automation and Artificial Intelligence (RAAI)*, IEEE, 2021, 69–72.

[86] R. Chandra Shit and S. Sharma, Ray-tracing assisted fingerprinting for localization in IoT Health 4.0, *Transactions on Emerging Telecommunications Technologies*, vol. 33, no. 11, e4573, 2022.

[87] V.-R. Xefteris, A. Tsanousa, G. Meditskos, S. Vrochidis and I. Kompatsiaris, Performance, challenges, and limitations in multimodal fall detection systems: a review, *IEEE Sensors Journal*, 2021.

[88] G. Baldini, R. Giuliani, G. Steri, I. Sanchez and C. Gentile, The application of the symbolic aggregate approximation algorithm (SAX) to radio frequency fingerprinting of IoT devices, in *2017 IEEE Symposium on Communications and Vehicular Technology (SCVT)*, IEEE, 2017, 1–6.

[89] S.-H. Cha, Comprehensive survey on distance/similarity measures between probability density functions, *City*, vol. 1, no. 2, 1, 2007.

[90] J. Torres-Sospedra, R. Montoliu, S. Trilles, Ó. Belmonte and J. Huerta, Comprehensive analysis of distance and similarity measures for Wi-Fi fingerprinting indoor positioning systems, *Expert Systems with Applications*, vol. 42, no. 23, 9263–9278, 2015.

[91] M. Simić-Pejović and A. Garaj, A comparative analysis of signal space distance metrics for fingerprinting based indoor positioning, in *2021 20th International Symposium INFOTEH-JAHORINA (INFOTEH)*, IEEE, 2021, 1–6.

[92] V. Moghtadaiee and A. G. Dempster, Vector distance measure comparison in indoor location fingerprinting, in *International Global Navigation Satellite Systems Society (IGNSS Symposium)*, 2015.

[93] J. D. Gutiérrez, A. R. Jiménez, F. Seco, F. J. Álvarez, T. Aguilera, J. Torres-Sospedra and F. Melchor, GetSensorData: An extensible Android-based application for multi-sensor data registration, *SoftwareX*, vol. 19, 101186, 2022.

[94] A. Moreira, I. Silva and J. Torres-Sospedra, The DSI dataset for Wi-Fi fingerprinting using mobile devices, version 1.0, version 1.0, *Zenodo, Apr*, 2020. DOI: 10.5281/zenodo.3778646. [Online]. Available: https://doi.org/10.5281/zenodo.3778646.

[95] G. M. Mendoza-Silva, P. Richter, J. Torres-Sospedra, E. S. Lohan and J. Huerta, Long-Term WiFi Fingerprinting Dataset for Research on Robust Indoor Positioning, *Data*, vol. 3, no. 1, 2018.

[96] T. King, S. Kopf, T. Haenselmann, C. Lubberger and W. Effelsberg, *CRAWDAD dataset mannheim/compass (v. 2008-04-11)*, Downloaded from https://crawdad.org/mannheim/compass/20080411, Apr. 2008.

[97] T. King, T. Haenselmann and W. Effelsberg, On-demand fingerprint selection for 802.11-based positioning systems, in *2008 International Symposium on a World of Wireless, Mobile and Multimedia Networks*, Jun. 2008, 1–8.

[98] A. Moreira, I. Silva, F. Meneses, M. J. Nicolau, C. Pendao and J. Torres-Sospedra, Multiple simultaneous Wi-Fi measurements in fingerprinting indoor positioning, in *Int. Conf. on Indoor Positioning and Indoor Navigation*, 2017.

[99] E. S. Lohan, J. Torres-Sospedra and A. Gonzalez, *WiFi RSS measurements in Tampere University multi- building campus, 2017*, version 1, Zenodo, Aug. 2021. DOI: 10.5281/zenodo.5174851. [Online]. Available: https://doi.org/10.5281/zenodo.5174851.

[100] A. Razavi, M. Valkama and E.-S. Lohan, K-means fingerprint clustering for low-complexity floor estimation in indoor mobile localization, in *2015 IEEE Globecom Workshops*, 2015.

[101] A. Cramariuc, H. Huttunen and E. S. Lohan, Clustering benefits in mobile-centric WiFi positioning in multi-floor buildings, in *2016 International Conference on Localization and GNSS*, 2016.

[102] E.-S. Lohan, J. Torres-Sospedra, H. Leppäkoski, P. Richter, Z. Peng and J. Huerta, Wi-Fi Crowdsourced Fingerprinting Dataset for Indoor Positioning, *MDPI Data*, vol. 2, no. 4, 2017, ISSN: 2306-5729.

[103] P. Richter, E. S. Lohan and J. Talvitie. "WLAN (WiFi) RSS database for fingerprinting positioning." (Jan. 2018), [Online]. Available: `https://zenodo.org/record/1161525`.

[104] Lohan. "Additional TAU datasets for Wi-Fi fingerprinting- based positioning." version v1, 11.05.2020. (May 2020), [Online]. Available: `https://doi.org/10.5281/zenodo.3819917`.

[105] F. J. Aranda, F. Parralejo, F. J. Álvarez and J. Torres-Sospedra, Multi-slot BLE raw database for accurate positioning in mixed indoor/outdoor environments, *Data*, vol. 5, no. 3, 67, 2020.

[106] J. Torres-Sospedra, R. Montoliu, A. Martínez-Usó, J. P. Avariento, T. J. Arnau, M. Benedito-Bordonau and J. Huerta, UJIIndoorLoc: A new multi-building and multi-floor database for WLAN fingerprint-based indoor localization problems, in *Int. Conf. on Indoor Positioning and Indoor Navigation*, 2014, 261–270.

[107] G. M. Mendoza-Silva, M. Matey-Sanz, J. Torres-Sospedra and J. Huerta, BLE RSS measurements dataset for research on accurate indoor positioning, *Data*, vol. 4, no. 1, 12, 2019.

[108] X. Song, X. Fan, C. Xiang, Q. Ye, L. Liu, Z. Wang, X. He, N. Yang and G. Fang, A novel convolutional neural network based indoor localization framework with WiFi fingerprinting, *IEEE Access*, vol. 7, 110698–110709, 2019.

[109] A. Saxena, M. Prasad, A. Gupta, N. Bharill, O. P. Patel, A. Tiwari, M. J. Er, W. Ding and C.-T. Lin, A review of clustering techniques and developments, *Neurocomputing*, vol. 267, 664–681, 2017.

[110] A. Anuwatkun, J. Sangthong and S. Sang-Ngern, A diff-based indoor positioning system using fingerprinting technique and k-means clustering algorithm, in *2019 16th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, IEEE, 2019, 148–151.

[111] Y. Cui, S. Gao and Y. Zheng, Application of ZigBee Location Fingerprint Method in Positioning of Railway Tunnel Staff, in *2018 Chinese Automation Congress (CAC)*, IEEE, 2018, 3283–3287.

[112] K. Yang, Q. Luo and X. Yan, Trilateration Based on the Combination and K-Means Clustering, in *2021 Global Reliability and Prognostics and Health Management (PHM-Nanjing)*, IEEE, 2021, 1–4.

[113] C. E. Rasmussen, Gaussian processes in machine learning, in *Summer school on machine learning*, Springer, 2003, 63–71.

[114] IEEE, IEEE Standard for Information technology-Telecommunication and information exchange between systems-Local and metropolitan area networks-Specific requirements Part11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment1: Radio Resource Measurement of Wireless LANs, 2009.

[115] ETSI, Wireless Access Systems (WAS); 5,8 GHz fixed broadband data transmitting systems; Harmonised Standard covering the essential requirements of article 3.2 of Directive 2014/53/EU, 2017.

[116] ETSI, Wideband transmission systems; Data transmission equipment operating in the 2,4 GHz band; Harmonized Standard for access to radio spectrum, 2019.

[117] P. Fränti and S. Sieranoja, How much can k-means be improved by using better initialization and repeats? *Pattern Recognition*, vol. 93, 95–112, 2019.

[118] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," Stanford, Tech. Rep., 2006.

[119] T. F. Gonzalez, Clustering to minimize the maximum intercluster distance, *Theoretical computer science*, vol. 38, 293–306, 1985.

[120] H. K. Palo, S. Sahoo and A. K. Subudhi, Dimensionality reduction techniques: Principles, benefits, and limitations, *Data Analytics in Bioinformatics: A Machine Learning Perspective*, 77–107, 2021.

[121] W. Chen, Y. Xu, Z. Yu, W. Cao, C. P. Chen and G. Han, Hybrid dimensionality reduction forest with pruning for high-dimensional data classification, *IEEE Access*, vol. 8, 40138–40150, 2020.

[122] A. Kushki, K. N. Plataniotis and A. N. Venetsanopoulos, Kernel-based positioning in wireless local area networks, *IEEE transactions on mobile computing*, vol. 6, no. 6, 689–705, 2007.

[123] I. T. Jolliffe and J. Cadima, Principal component analysis: a review and recent developments, *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 374, no. 2065, 20150202, 2016.

[124] M. A. Kramer, Nonlinear principal component analysis using autoassociative neural networks, *AIChE journal*, vol. 37, no. 2, 233–243, 1991.

[125] H. Abdi and L. J. Williams, Principal component analysis, *Wiley interdisciplinary reviews: computational statistics*, vol. 2, no. 4, 433–459, 2010.

[126] X. Zhao, J. Guo, F. Nie, L. Chen, Z. Li and H. Zhang, Joint principal component and discriminant analysis for dimensionality reduction, *IEEE Trans. on neural networks and learning systems*, vol. 31, no. 2, 433–444, 2019.

[127] Z. Xia, Y. Chen and C. Xu, Multiview PCA: A Methodology of Feature Extraction and Dimension Reduction for High-Order Data, *IEEE Trans. on Cybernetics*, 2021.

[128] A. Li, J. Fu, H. Shen and S. Sun, A cluster-principal-component-analysis-based indoor positioning algorithm, *IEEE Internet of Things Journal*, vol. 8, no. 1, 187–196, 2020.

[129] B. Chidlovskii and L. Antsfeld, Semi-supervised variational autoencoder for Wi-Fi indoor localization, in *2019 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, IEEE, 2019, 1–8.

[130] G. JunLin, Z. Xin, W. HuaDeng and Y. Lan, WiFi fingerprint positioning method based on fusion of autoencoder and stacking mode, in *2020 International Conference on Culture-Oriented Science & Technology (ICCST)*, IEEE, 2020, 356–361.

[131] G. Li, S. Peng, C. Wang, J. Niu and Y. Yuan, An energy-efficient data collection scheme using denoising autoencoder in wireless sensor networks, *Tsinghua Science and Technology*, vol. 24, no. 1, 86–96, 2018.

[132] L. Hou, V. Nguyen, A. B. Kanevsky, D. Samaras, T. M. Kurc, T. Zhao, R. R. Gupta, Y. Gao, W. Chen, D. Foran *et al.*, Sparse autoencoder for unsupervised nucleus detection and representation in histopathology images, *Pattern recognition*, vol. 86, 188–200, 2019.

[133] A. Ng *et al.*, Sparse autoencoder, *CS294A Lecture notes*, vol. 72, no. 2011, 1–19, 2011.

[134] Machine Learning Explained: Regularization, *Enhance Data ScienceS*, 2017. [Online]. Available: `http://enhancedatascience.com/2017/07/04/machine-learning-explained-regularization/`.

[135] Z. Chen, C. K. Yeo, B. S. Lee and C. T. Lau, Autoencoder-based network anomaly detection, in *2018 Wireless Telecommunications Symposium (WTS)*, IEEE, 2018, 1–5.

[136] L. Zhao, H. Huang, X. Li, S. Ding, H. Zhao and Z. Han, An accurate and robust approach of device-free localization with convolutional autoencoder, *IEEE Internet of Things Journal*, vol. 6, no. 3, 5825–5840, 2019.

[137] J. Torres-Sospedra, D. Quezada-Gaibor, G. M. Mendoza-Silva, J. Nurmi, Y. Koucheryavy and J. Huerta, New cluster selection and fine-grained search for k-means clustering and Wi-Fi fingerprinting, in *2020 International Conference on Localization and GNSS (ICL-GNSS)*, IEEE, 2020, 1–6.

[138] M. A. Bhatti, R. Riaz, S. S. Rizvi, S. Shokat, F. Riaz and S. J. Kwon, Outlier detection in indoor localization and Internet of Things (IoT) using machine learning, *Journal of Communications and Networks*, vol. 22, no. 3, 236–243, 2020.

[139] B. Ezhumalai, M. Song and K. Park, An efficient indoor positioning method based on Wi-Fi RSS fingerprint and classification algorithm, *Sensors*, vol. 21, no. 10, 3418, 2021.

[140] J. Rojo, G. M. Mendoza-Silva, G. R. Cidral, J. Laiapea, G. Parrello, A. Simó, L. Stupin, D. Minican, M. Farrés, C. Corvalán *et al.*, Machine learning applied to Wi-Fi fingerprinting: The experiences of the ubiqum challenge, in *2019 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, IEEE, 2019, 1–8.

[141] S. Premchaisawatt and N. Ruangchaijatupon, Enhancing indoor positioning based on partitioning cascade machine learning models, in *2014 11th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, IEEE, 2014, 1–5.

[142] R.-C. Chen, Y.-C. Lin and Y.-S. Lin, Indoor position location based on cascade correlation networks, in *2011 IEEE International Conference on Systems, Man, and Cybernetics*, IEEE, 2011, 2295–2300.

[143] M. N. Borenović and A. M. Nešković, Positioning in WLAN environment by use of artificial neural networks and space partitioning, *annals of telecommunications-annales des télécommunications*, vol. 64, no. 9, 665–676, 2009.

[144] T. Hastie, S. Rosset, J. Zhu and H. Zou, Multi-class adaboost, *Statistics and its Interface*, vol. 2, no. 3, 349–360, 2009.

[145] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, *arXiv preprint arXiv:1412.6980*, 2014.

**APPENDIX A**

# ADDITIONS TO 4.1

**Table A.1** 2D positioning regression results - Full validation [m]

Begin of Table A.1

| Dataset | 1NN | W3NN | DT | RFor | AdB | NN *simple* | NN 10 | NN 100 | NN [100,100] |
|---|---|---|---|---|---|---|---|---|---|
| LIB $1_A$ | 0.16 | 0.54 | 1.41 | 1.42 | 2.12 | 2.14 | 1.57 | 1.58 | 1.39 |
| LIB $1_B$ | 0.51 | 0.80 | 1.29 | 1.37 | 1.78 | 2.60 | 1.85 | 1.62 | 1.51 |
| LIB $2_A$ | 0.22 | 0.58 | 1.58 | 1.66 | 1.98 | 1.97 | 1.82 | 1.48 | 1.19 |
| LIB $2_B$ | 0.39 | 0.97 | 1.74 | 1.77 | 2.16 | 2.45 | 2.07 | 1.79 | 1.62 |
| SAH $1_A$ | 0.84 | 0.86 | 1.51 | 1.27 | 16.44 | 11.18 | 3.87 | 2.27 | 1.83 |
| SAH $1_B$ | 0.69 | 0.71 | 1.67 | 1.32 | 19.02 | 9.02 | 2.79 | 1.77 | 1.60 |
| SAH $1_C$ | 0.91 | 0.88 | 2.50 | 1.53 | 13.65 | 8.10 | 3.51 | 2.02 | 1.68 |
| TIE $1_A$ | 0.64 | 0.68 | 1.41 | 1.29 | 1.80 | 5.80 | 3.60 | 1.78 | 1.49 |
| TIE $1_B$ | 0.69 | 0.69 | 1.84 | 1.44 | 14.56 | 8.95 | 3.65 | 2.10 | 1.73 |
| TIE $1_C$ | 0.63 | 0.64 | 1.85 | 1.56 | 5.08 | 6.73 | 2.82 | 1.67 | 1.38 |
| TIE $1_D$ | 0.56 | 0.54 | 1.65 | 1.18 | 3.01 | 5.20 | 3.24 | 1.81 | 1.55 |

Continuation of Table A.1

| Dataset | 1NN | W3NN | DT | RFor | AdB | NN *simple* | NN 10 | NN 100 | NN [100,100] |
|---|---|---|---|---|---|---|---|---|---|
| TIE $1_E$ | 0.69 | 0.71 | 1.51 | 1.13 | 2.94 | 6.40 | 3.63 | 1.68 | 1.22 |
| TIE $1_F$ | 1.48 | 1.34 | 1.21 | 1.26 | 1.27 | 2.04 | 2.32 | 1.63 | 1.56 |
| TUT $1_A$ | 5.51 | 4.55 | 8.76 | 4.77 | 8.37 | 10.75 | 7.04 | 5.48 | 5.16 |
| TUT $1_B$ | 5.32 | 4.65 | 8.32 | 4.87 | 7.10 | 10.88 | 6.91 | 5.67 | 4.84 |
| TUT $1_C$ | 4.24 | 5.21 | 8.00 | 8.74 | 7.72 | 16.95 | 7.83 | 6.59 | 5.24 |
| TUT $1_D$ | 4.86 | 4.49 | 10.77 | 6.53 | 5.89 | 17.52 | 7.66 | 5.85 | 6.10 |
| TUT $2_A$ | 6.79 | 6.70 | 6.67 | 8.30 | 11.70 | 30.33 | 16.03 | 8.34 | 7.30 |
| TUT $2_B$ | 6.41 | 5.92 | 6.31 | 5.94 | 8.33 | 21.42 | 10.31 | 7.15 | 5.98 |
| TUT $2_C$ | 5.71 | 6.55 | 10.17 | 5.71 | 7.25 | 19.97 | 8.64 | 5.80 | 5.68 |
| TUT $3_A$ | 12.64 | 13.00 | 13.88 | 11.56 | 13.55 | 19.34 | 16.54 | 14.53 | 14.23 |
| TUT $3_B$ | 12.46 | 10.68 | 15.16 | 9.14 | 11.29 | 14.15 | 10.07 | 10.00 | 9.55 |
| TUT $3_C$ | 14.30 | 14.90 | 11.29 | 11.02 | 14.84 | 18.65 | 13.11 | 15.59 | 13.37 |
| TUT $3_D$ | 10.20 | 12.07 | 10.70 | 11.09 | 11.65 | 18.09 | 14.05 | 11.85 | 9.41 |
| TUT $3_E$ | 3.54 | 16.59 | 20.35 | 19.37 | 3.92 | 35.70 | 25.44 | 10.11 | 9.43 |
| TUT $4_A$ | 6.16 | 5.37 | 8.12 | 4.92 | 12.04 | 9.78 | 6.44 | 5.63 | 5.50 |
| TUT $4_B$ | 7.10 | 6.58 | 9.02 | 6.15 | 11.80 | 10.91 | 7.95 | 7.89 | 7.17 |
| TUT $4_C$ | 6.90 | 6.95 | 7.76 | 6.50 | 14.41 | 13.15 | 9.26 | 7.38 | 7.01 |
| TUT $4_D$ | 5.87 | 5.53 | 8.21 | 6.40 | 11.97 | 15.40 | 9.15 | 7.36 | 6.29 |
| TUT $4_E$ | 2.90 | 4.98 | 2.53 | 4.43 | 3.45 | 31.47 | 23.13 | 11.88 | 7.66 |
| TUT $5_A$ | 6.11 | 5.12 | 7.89 | 8.11 | 7.92 | 27.95 | 10.18 | 6.51 | 5.97 |
| TUT $5_B$ | 5.66 | 4.78 | 7.09 | 5.36 | 11.31 | 23.84 | 8.58 | 5.79 | 4.68 |
| TUT $5_C$ | 7.17 | 5.69 | 8.36 | 7.07 | 9.15 | 23.26 | 9.69 | 6.64 | 5.68 |
| TUT $6_A$ | 2.81 | 3.12 | 5.57 | 3.94 | 19.32 | 12.59 | 6.69 | 5.08 | 4.30 |
| TUT $6_B$ | 1.89 | 2.37 | 5.34 | 3.76 | 6.85 | 10.60 | 5.25 | 3.63 | 3.15 |
| TUT $6_C$ | 2.62 | 2.57 | 3.37 | 3.51 | 4.18 | 7.05 | 5.20 | 3.62 | 2.76 |
| TUT $6_D$ | 1.98 | 2.54 | 2.46 | 3.29 | 4.29 | 10.48 | 6.42 | 3.73 | 2.70 |
| TUT $7_A$ | 3.57 | 3.81 | 5.32 | 4.45 | 18.77 | 18.47 | 9.73 | 7.20 | 5.26 |
| TUT $7_B$ | 1.90 | 2.15 | 4.01 | 3.12 | 23.31 | 11.45 | 5.32 | 3.72 | 3.16 |

Continuation of Table A.1

| Dataset | 1NN | W3NN | DT | RFor | AdB | NN *simple* | NN 10 | NN 100 | NN [100,100] |
|---|---|---|---|---|---|---|---|---|---|
| TUT $7_C$ | 2.21 | 2.85 | 4.02 | 3.41 | 21.06 | 14.14 | 8.45 | 4.75 | 3.11 |
| UJI $1_1A$ | 0.90 | 1.23 | 3.20 | 3.38 | 6.81 | 13.69 | 6.74 | 4.84 | 3.57 |
| UJI $1_1B$ | 0.96 | 1.24 | 2.34 | 2.57 | 6.90 | 11.46 | 5.38 | 3.93 | 3.00 |
| UJI $1_1C$ | 2.07 | 2.18 | 2.53 | 2.94 | 6.00 | 10.32 | 5.10 | 3.95 | 3.49 |
| UJI $1_1D$ | 2.48 | 2.32 | 2.86 | 2.58 | 6.47 | 10.08 | 4.74 | 4.04 | 3.56 |
| UJI $1_2A$ | 1.56 | 1.90 | 3.45 | 3.51 | 9.62 | 13.38 | 7.68 | 5.09 | 4.15 |
| UJI $1_2B$ | 2.66 | 2.85 | 4.18 | 4.50 | 12.06 | 17.42 | 8.06 | 5.72 | 5.15 |
| UJI $1_2C$ | 0.79 | 1.28 | 3.62 | 3.33 | 10.01 | 12.65 | 6.37 | 4.40 | 3.47 |
| UJI $1_2D$ | 4.38 | 3.62 | 4.06 | 4.38 | 11.35 | 18.78 | 9.38 | 7.14 | 6.23 |
| UJI $1_3A$ | 1.45 | 1.66 | 3.43 | 3.73 | 9.03 | 12.52 | 7.96 | 5.32 | 3.53 |
| UJI $1_3B$ | 0.84 | 1.35 | 4.06 | 4.11 | 9.91 | 12.45 | 6.85 | 5.50 | 3.83 |
| UJI $1_3C$ | 0.70 | 1.10 | 3.78 | 3.58 | 7.90 | 10.37 | 6.70 | 4.64 | 3.44 |
| UJI $1_3D$ | 0.55 | 1.00 | 3.01 | 2.91 | 7.53 | 10.71 | 6.24 | 4.39 | 3.13 |
| UJI $1_3E$ | 6.36 | 5.18 | 5.89 | 5.94 | 9.67 | 11.41 | 8.10 | 7.33 | 6.85 |
| UJI $2_1A$ | 1.18 | 1.53 | 4.02 | 3.19 | 7.00 | 12.59 | 6.13 | 4.43 | 3.24 |
| UJI $2_1B$ | 1.96 | 1.89 | 3.21 | 3.10 | 7.49 | 11.59 | 5.78 | 4.28 | 3.42 |
| UJI $2_1C$ | 2.52 | 2.68 | 3.47 | 2.99 | 6.25 | 10.46 | 5.28 | 4.42 | 4.04 |
| UJI $2_1D$ | 2.15 | 2.55 | 3.67 | 2.84 | 6.14 | 10.64 | 5.21 | 4.37 | 3.85 |
| UJI $2_2A$ | 2.14 | 2.57 | 3.64 | 3.76 | 10.17 | 14.02 | 7.81 | 5.43 | 4.48 |
| UJI $2_2B$ | 2.89 | 3.18 | 4.49 | 4.44 | 12.32 | 16.86 | 8.02 | 6.22 | 5.51 |
| UJI $2_2C$ | 1.42 | 1.73 | 4.05 | 3.75 | 10.84 | 12.84 | 7.00 | 5.16 | 4.30 |
| UJI $2_2D$ | 1.75 | 2.52 | 4.26 | 4.10 | 11.77 | 18.18 | 9.53 | 7.25 | 5.79 |
| UJI $2_3A$ | 1.66 | 1.58 | 3.76 | 3.96 | 9.13 | 12.45 | 8.12 | 5.38 | 3.49 |
| UJI $2_3B$ | 1.47 | 1.82 | 5.22 | 4.58 | 9.32 | 12.98 | 7.53 | 6.13 | 4.59 |
| UJI $2_3C$ | 1.13 | 1.41 | 3.79 | 3.85 | 8.11 | 10.36 | 6.35 | 4.79 | 3.66 |
| UJI $2_3D$ | 0.91 | 1.16 | 3.57 | 3.15 | 7.57 | 10.94 | 6.25 | 4.36 | 3.04 |
| UJI $2_3E$ | 5.43 | 5.17 | 5.79 | 5.86 | 10.42 | 11.32 | 7.88 | 7.01 | 6.73 |
| UTS $1_A$ | 0.00 | 0.04 | 0.04 | 0.77 | 0.57 | 2.59 | 1.99 | 0.22 | 0.12 |

Continuation of Table A.1

| Dataset | 1NN | W3NN | DT | RFor | AdB | NN *simple* | NN 10 | NN 100 | NN [100,100] |
|---|---|---|---|---|---|---|---|---|---|
| UTS $1_B$ | 0.33 | 0.33 | 0.45 | 1.16 | 1.33 | 3.14 | 2.66 | 0.74 | 0.19 |
| UTS $1_C$ | 0.51 | 0.72 | 0.63 | 1.29 | 2.52 | 6.19 | 4.93 | 2.48 | 1.38 |
| UTS $1_D$ | 0.04 | 0.05 | 0.17 | 0.37 | 4.11 | 8.13 | 3.66 | 1.10 | 0.58 |
| UTS $1_E$ | 0.31 | 0.17 | 0.19 | 0.52 | 2.91 | 5.65 | 4.24 | 1.04 | 0.48 |
| UTS $1_F$ | 0.03 | 0.04 | 0.46 | 0.49 | 2.55 | 4.36 | 2.49 | 0.45 | 0.28 |
| UTS $1_G$ | 0.08 | 0.08 | 0.19 | 0.30 | 2.15 | 6.48 | 3.32 | 0.57 | 0.09 |
| UTS $1_H$ | 0.13 | 0.17 | 0.15 | 0.33 | 3.97 | 6.45 | 2.21 | 0.60 | 0.29 |
| UTS $1_I$ | 0.23 | 0.17 | 0.34 | 0.83 | 3.00 | 7.86 | 2.88 | 1.31 | 0.76 |
| UTS $1_J$ | 0.24 | 0.28 | 0.71 | 1.00 | 3.15 | 6.85 | 3.51 | 1.52 | 0.77 |
| UTS $1_K$ | 0.30 | 0.39 | 1.28 | 2.88 | 5.34 | 13.78 | 8.17 | 4.22 | 1.33 |
| UTS $1_L$ | 0.62 | 1.00 | 0.87 | 1.60 | 4.68 | 11.63 | 5.87 | 3.22 | 1.49 |
| UTS $1_M$ | 0.01 | 0.01 | 0.02 | 0.11 | 3.55 | 7.83 | 3.50 | 1.03 | 0.17 |
| UTS $1_N$ | 0.17 | 0.18 | 0.20 | 0.26 | 2.61 | 4.74 | 2.17 | 0.77 | 0.62 |
| UTS $1_O$ | 0.14 | 0.11 | 0.19 | 0.18 | 2.86 | 5.46 | 2.88 | 0.71 | 0.54 |
| UTS $1_P$ | 0.10 | 0.07 | 0.06 | 0.17 | 2.84 | 4.41 | 1.81 | 0.39 | 0.17 |
| Average | 2.69 | 2.93 | 4.22 | 3.77 | 7.99 | 11.99 | 6.69 | 4.60 | 3.77 |

End of Table A.1

**Table A.2** 2D positioning regression results - Full validation of prediction time [s]

Begin of Table A.2

| Dataset | 1NN | W3NN | DT | RF | AdB | NN *simple* | NN 10 | NN 100 | NN [100,100] |
|---|---|---|---|---|---|---|---|---|---|
| LIB $1_A$ | 0.002 | 0.003 | 0.000 | 0.031 | 0.030 | 0.000 | 0.000 | 0.001 | 0.001 |
| LIB $1_B$ | 0.003 | 0.003 | 0.000 | 0.020 | 0.040 | 0.000 | 0.000 | 0.000 | 0.000 |
| LIB $2_A$ | 0.004 | 0.003 | 0.000 | 0.026 | 0.063 | 0.000 | 0.000 | 0.000 | 0.000 |
| LIB $2_B$ | 0.002 | 0.003 | 0.000 | 0.020 | 0.040 | 0.000 | 0.008 | 0.000 | 0.000 |
| SAH $1_A$ | 0.523 | 0.532 | 0.010 | 0.030 | 0.091 | 0.000 | 0.000 | 0.001 | 0.008 |

Continuation of Table A.2

| Dataset | 1NN | W3NN | DT | RF | AdB | NN *simple* | NN 10 | NN 100 | NN [100,100] |
|---|---|---|---|---|---|---|---|---|---|
| SAH $1_B$ | 3.932 | 3.944 | 0.013 | 0.043 | 0.568 | 0.000 | 0.000 | 0.000 | 0.000 |
| SAH $1_C$ | 0.568 | 0.582 | 0.008 | 0.020 | 0.203 | 0.000 | 0.000 | 0.000 | 0.002 |
| TIE $1_A$ | 0.003 | 0.004 | 0.000 | 0.017 | 0.031 | 0.000 | 0.000 | 0.000 | 0.000 |
| TIE $1_B$ | 3.907 | 3.874 | 0.010 | 0.043 | 0.286 | 0.008 | 0.002 | 0.010 | 0.000 |
| TIE $1_C$ | 0.648 | 0.647 | 0.000 | 0.023 | 0.299 | 0.000 | 0.000 | 0.000 | 0.008 |
| TIE $1_D$ | 0.100 | 0.099 | 0.000 | 0.020 | 0.092 | 0.000 | 0.000 | 0.000 | 0.000 |
| TIE $1_E$ | 0.085 | 0.086 | 0.002 | 0.020 | 0.101 | 0.000 | 0.000 | 0.000 | 0.001 |
| TIE $1_F$ | 0.001 | 0.001 | 0.000 | 0.022 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| TUT $1_A$ | 0.023 | 0.023 | 0.001 | 0.020 | 0.053 | 0.000 | 0.000 | 0.000 | 0.000 |
| TUT $1_B$ | 0.017 | 0.018 | 0.000 | 0.020 | 0.051 | 0.001 | 0.002 | 0.008 | 0.000 |
| TUT $1_C$ | 0.001 | 0.001 | 0.008 | 0.018 | 0.033 | 0.000 | 0.000 | 0.000 | 0.000 |
| TUT $1_D$ | 0.000 | 0.001 | 0.000 | 0.018 | 0.028 | 0.000 | 0.001 | 0.000 | 0.000 |
| TUT $2_A$ | 0.002 | 0.002 | 0.001 | 0.016 | 0.012 | 0.000 | 0.000 | 0.000 | 0.000 |
| TUT $2_B$ | 0.004 | 0.004 | 0.000 | 0.031 | 0.032 | 0.002 | 0.000 | 0.000 | 0.000 |
| TUT $2_C$ | 0.001 | 0.002 | 0.000 | 0.017 | 0.030 | 0.000 | 0.000 | 0.000 | 0.000 |
| TUT $3_A$ | 0.012 | 0.012 | 0.000 | 0.012 | 0.051 | 0.000 | 0.000 | 0.000 | 0.001 |
| TUT $3_B$ | 0.008 | 0.007 | 0.000 | 0.020 | 0.040 | 0.000 | 0.000 | 0.000 | 0.000 |
| TUT $3_C$ | 0.004 | 0.003 | 0.000 | 0.017 | 0.040 | 0.000 | 0.000 | 0.000 | 0.000 |
| TUT $3_D$ | 0.003 | 0.003 | 0.000 | 0.020 | 0.041 | 0.000 | 0.008 | 0.000 | 0.000 |
| TUT $3_E$ | 0.000 | 0.000 | 0.000 | 0.016 | 0.031 | 0.000 | 0.000 | 0.000 | 0.000 |
| TUT $4_A$ | 0.326 | 0.311 | 0.000 | 0.030 | 0.164 | 0.000 | 0.000 | 0.000 | 0.000 |
| TUT $4_B$ | 0.218 | 0.225 | 0.000 | 0.020 | 0.150 | 0.000 | 0.000 | 0.001 | 0.000 |
| TUT $4_C$ | 0.102 | 0.100 | 0.000 | 0.015 | 0.111 | 0.000 | 0.002 | 0.008 | 0.000 |
| TUT $4_D$ | 0.083 | 0.083 | 0.000 | 0.020 | 0.102 | 0.000 | 0.000 | 0.000 | 0.000 |
| TUT $4_E$ | 0.002 | 0.002 | 0.008 | 0.018 | 0.028 | 0.000 | 0.000 | 0.000 | 0.000 |
| TUT $5_A$ | 0.001 | 0.002 | 0.000 | 0.016 | 0.021 | 0.000 | 0.000 | 0.000 | 0.000 |
| TUT $5_B$ | 0.004 | 0.004 | 0.000 | 0.018 | 0.038 | 0.000 | 0.000 | 0.000 | 0.000 |
| TUT $5_C$ | 0.001 | 0.001 | 0.000 | 0.018 | 0.028 | 0.000 | 0.000 | 0.000 | 0.000 |

Continuation of Table A.2

| Dataset | 1NN | W3NN | DT | RF | AdB | NN *simple* | NN 10 | NN 100 | NN [100,100] |
|---|---|---|---|---|---|---|---|---|---|
| TUT $6_A$ | 0.426 | 0.380 | 0.000 | 0.030 | 0.102 | 0.000 | 0.000 | 0.000 | 0.002 |
| TUT $6_B$ | 0.061 | 0.060 | 0.008 | 0.030 | 0.076 | 0.000 | 0.000 | 0.000 | 0.000 |
| TUT $6_C$ | 0.009 | 0.010 | 0.008 | 0.020 | 0.027 | 0.000 | 0.000 | 0.000 | 0.000 |
| TUT $6_D$ | 0.010 | 0.009 | 0.000 | 0.018 | 0.040 | 0.000 | 0.000 | 0.000 | 0.000 |
| TUT $7_A$ | 0.067 | 0.060 | 0.001 | 0.027 | 0.010 | 0.000 | 0.000 | 0.008 | 0.000 |
| TUT $7_B$ | 0.371 | 0.366 | 0.000 | 0.020 | 0.081 | 0.000 | 0.000 | 0.000 | 0.000 |
| TUT $7_C$ | 0.044 | 0.045 | 0.000 | 0.019 | 0.020 | 0.000 | 0.000 | 0.000 | 0.000 |
| UJI $1_1A$ | 0.094 | 0.094 | 0.001 | 0.002 | 0.091 | 0.000 | 0.000 | 0.002 | 0.008 |
| UJI $1_1B$ | 0.153 | 0.156 | 0.000 | 0.016 | 0.101 | 0.000 | 0.000 | 0.003 | 0.000 |
| UJI $1_1C$ | 0.175 | 0.177 | 0.000 | 0.010 | 0.103 | 0.000 | 0.000 | 0.000 | 0.000 |
| UJI $1_1D$ | 0.161 | 0.162 | 0.002 | 0.010 | 0.098 | 0.000 | 0.000 | 0.002 | 0.000 |
| UJI $1_2A$ | 0.155 | 0.157 | 0.008 | 0.017 | 0.069 | 0.000 | 0.000 | 0.000 | 0.000 |
| UJI $1_2B$ | 0.182 | 0.184 | 0.000 | 0.018 | 0.031 | 0.002 | 0.008 | 0.001 | 0.008 |
| UJI $1_2C$ | 0.162 | 0.162 | 0.000 | 0.010 | 0.030 | 0.000 | 0.000 | 0.001 | 0.008 |
| UJI $1_2D$ | 0.069 | 0.069 | 0.000 | 0.010 | 0.012 | 0.002 | 0.000 | 0.008 | 0.000 |
| UJI $1_3A$ | 0.317 | 0.311 | 0.002 | 0.010 | 0.141 | 0.000 | 0.000 | 0.000 | 0.000 |
| UJI $1_3B$ | 0.401 | 0.404 | 0.000 | 0.008 | 0.159 | 0.000 | 0.002 | 0.000 | 0.000 |
| UJI $1_3C$ | 0.208 | 0.211 | 0.000 | 0.010 | 0.121 | 0.000 | 0.000 | 0.000 | 0.000 |
| UJI $1_3D$ | 0.651 | 0.648 | 0.002 | 0.013 | 0.307 | 0.000 | 0.000 | 0.008 | 0.008 |
| UJI $1_3E$ | 0.100 | 0.101 | 0.001 | 0.010 | 0.051 | 0.000 | 0.000 | 0.000 | 0.000 |
| UJI $2_1A$ | 0.107 | 0.108 | 0.001 | 0.010 | 0.064 | 0.001 | 0.000 | 0.001 | 0.001 |
| UJI $2_1B$ | 0.208 | 0.205 | 0.001 | 0.005 | 0.125 | 0.000 | 0.000 | 0.000 | 0.000 |
| UJI $2_1C$ | 0.217 | 0.219 | 0.000 | 0.012 | 0.123 | 0.000 | 0.000 | 0.000 | 0.000 |
| UJI $2_1D$ | 0.183 | 0.189 | 0.000 | 0.010 | 0.115 | 0.000 | 0.000 | 0.001 | 0.000 |
| UJI $2_2A$ | 0.162 | 0.165 | 0.000 | 0.010 | 0.064 | 0.000 | 0.000 | 0.002 | 0.002 |
| UJI $2_2B$ | 0.222 | 0.224 | 0.000 | 0.010 | 0.072 | 0.000 | 0.000 | 0.000 | 0.000 |
| UJI $2_2C$ | 0.182 | 0.184 | 0.002 | 0.008 | 0.031 | 0.000 | 0.000 | 0.000 | 0.001 |
| UJI $2_2D$ | 0.077 | 0.077 | 0.000 | 0.011 | 0.029 | 0.000 | 0.000 | 0.000 | 0.001 |

Continuation of Table A.2

| Dataset | 1NN | W3NN | DT | RF | AdB | NN *simple* | NN 10 | NN 100 | NN [100,100] |
|---|---|---|---|---|---|---|---|---|---|
| UJI $2_3A$ | 0.317 | 0.317 | 0.000 | 0.010 | 0.152 | 0.000 | 0.000 | 0.002 | 0.000 |
| UJI $2_3B$ | 0.449 | 0.446 | 0.002 | 0.020 | 0.164 | 0.000 | 0.000 | 0.002 | 0.000 |
| UJI $2_3C$ | 0.222 | 0.224 | 0.000 | 0.009 | 0.115 | 0.000 | 0.000 | 0.000 | 0.000 |
| UJI $2_3D$ | 0.671 | 0.672 | 0.008 | 0.018 | 0.317 | 0.000 | 0.000 | 0.000 | 0.002 |
| UJI $2_3E$ | 0.108 | 0.108 | 0.000 | 0.010 | 0.051 | 0.000 | 0.000 | 0.000 | 0.001 |
| UTS $1_A$ | 0.002 | 0.001 | 0.001 | 0.023 | 0.042 | 0.000 | 0.001 | 0.001 | 0.001 |
| UTS $1_B$ | 0.001 | 0.001 | 0.002 | 0.021 | 0.043 | 0.000 | 0.001 | 0.000 | 0.001 |
| UTS $1_C$ | 0.009 | 0.010 | 0.001 | 0.029 | 0.091 | 0.001 | 0.000 | 0.001 | 0.001 |
| UTS $1_D$ | 0.051 | 0.051 | 0.001 | 0.031 | 0.097 | 0.001 | 0.000 | 0.001 | 0.001 |
| UTS $1_E$ | 0.017 | 0.017 | 0.002 | 0.027 | 0.046 | 0.001 | 0.000 | 0.000 | 0.001 |
| UTS $1_F$ | 0.019 | 0.020 | 0.001 | 0.024 | 0.079 | 0.001 | 0.000 | 0.001 | 0.000 |
| UTS $1_G$ | 0.024 | 0.024 | 0.001 | 0.025 | 0.100 | 0.000 | 0.001 | 0.001 | 0.001 |
| UTS $1_H$ | 0.114 | 0.119 | 0.002 | 0.035 | 0.128 | 0.001 | 0.001 | 0.001 | 0.001 |
| UTS $1_I$ | 0.020 | 0.020 | 0.002 | 0.026 | 0.079 | 0.001 | 0.001 | 0.001 | 0.001 |
| UTS $1_J$ | 0.023 | 0.023 | 0.001 | 0.022 | 0.070 | 0.001 | 0.001 | 0.001 | 0.001 |
| UTS $1_K$ | 0.017 | 0.017 | 0.002 | 0.021 | 0.046 | 0.000 | 0.001 | 0.001 | 0.001 |
| UTS $1_L$ | 0.021 | 0.021 | 0.001 | 0.021 | 0.070 | 0.000 | 0.001 | 0.001 | 0.001 |
| UTS $1_M$ | 0.080 | 0.080 | 0.002 | 0.024 | 0.078 | 0.000 | 0.001 | 0.002 | 0.001 |
| UTS $1_N$ | 0.087 | 0.088 | 0.003 | 0.027 | 0.071 | 0.000 | 0.000 | 0.002 | 0.001 |
| UTS $1_O$ | 0.053 | 0.054 | 0.004 | 0.036 | 0.118 | 0.000 | 0.001 | 0.001 | 0.001 |
| UTS $1_P$ | 0.079 | 0.079 | 0.003 | 0.028 | 0.146 | 0.000 | 0.001 | 0.001 | 0.001 |
| Average | 0.221 | 0.221 | 0.002 | 0.019 | 0.089 | 0.000 | 0.001 | 0.001 | 0.001 |

End of Table A.2

**APPENDIX B**

# ADDITIONS TO 4.2

**Table B.1** Overall performance of considered compression techniques

Begin of Table B.1

| Dataset | CR | $\tilde{\zeta}_{\mathcal{B}\alpha}$ [-] | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] |
|---|---|---|---|---|---|---|
| | | | **EWO8** | | | |
| DSI 1 | 2.33 | - | - | 1.02 | 1.02 | 1 |
| LIB 1 | 2.33 | - | 1.00 | 1.01 | 1.01 | 1 |
| LIB 2 | 2.33 | - | 1.00 | 1.04 | 1.04 | 1 |
| MAN 1 | 2.33 | - | - | 1.08 | 1.08 | 1 |
| MAN 2 | 2.33 | - | - | 0.98 | 0.98 | 1 |
| MINT 1 | 2.33 | - | - | 1.08 | 1.08 | 1 |
| SIM 1 | 2.33 | - | - | 1.10 | 1.10 | 1 |
| TUJI 1 | 2.33 | - | - | 1.05 | 1.05 | 1 |
| UJI 1 | 2.33 | 1.00 | 1.00 | 1.00 | 1.01 | 1 |

Continuation of Table B.1

| Dataset | CR | $\tilde{\zeta}_{\mathcal{B}\alpha}$ [-] | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] |
|---|---|---|---|---|---|---|
| UJI 2 | 2.33 | 1.00 | 0.99 | 1.02 | 1.01 | 1 |
| UJIB 1 | 2.33 | - | - | 1.00 | 1.00 | 1 |
| UJIB 2 | 2.33 | - | - | 0.97 | 0.97 | 1 |
| UTS 1 | 2.33 | - | 0.99 | 1.06 | 1.06 | 1 |
| Average | 2.33 | 1.00 | 1.00 | 1.03 | 1.03 | 1.00 |
| | | | **PCA** | | | |
| DSI 1 | 9.2 | - | - | 1.18 | 1.18 | 0.08 |
| LIB 1 | 43.5 | - | 1.00 | 0.98 | 0.98 | 0.02 |
| LIB 2 | 32.8 | - | 0.99 | 1.47 | 1.45 | 0.12 |
| MAN 1 | 5.6 | - | - | 1.12 | 1.12 | 0.04 |
| MAN 2 | 14 | - | - | 1.33 | 1.33 | 0.04 |
| MINT 1 | 5.5 | - | - | 1.07 | 1.07 | 0.03 |
| SIM 1 | 2.7 | - | - | 1.04 | 1.04 | 0.03 |
| TUJI 1 | 15.5 | - | - | 1.16 | 1.16 | 0.18 |
| UJI 1 | 15.8 | 1.00 | 0.98 | 1.05 | 1.06 | 0.14 |
| UJI 2 | 15.3 | 1.00 | 1.00 | 1.02 | 1.07 | 0.17 |
| UJIB 1 | 2.2 | - | - | 1.05 | 1.05 | 0.11 |
| UJIB 2 | 1.8 | - | - | 1.07 | 1.07 | 0.09 |
| UTS 1 | 17.3 | - | 1.04 | 1.04 | 1.04 | 0.20 |
| Average | 13.94 | 1.00 | 1.00 | 1.12 | 1.12 | 0.09 |
| | | | **AE** | | | |
| DSI 1 | 9.2 | - | - | 1.25 | 1.25 | 0.08 |
| LIB 1 | 43.5 | - | 0.98 | 1.15 | 1.13 | 0.04 |

Continuation of Table B.1

| Dataset | CR | $\tilde{\zeta}_{\mathcal{B}\alpha}$ [-] | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] |
|---|---|---|---|---|---|---|
| LIB 2 | 32.8 | - | 0.99 | 1.69 | 1.67 | 0.11 |
| MAN 1 | 5.6 | - | - | 1.55 | 1.55 | 0.03 |
| MAN 2 | 14 | - | - | 1.75 | 1.75 | 0.07 |
| MINT 1 | 5.5 | - | - | 1.22 | 1.22 | 0.03 |
| SIM 1 | 2.7 | - | - | 1.32 | 1.32 | 0.04 |
| TUJI 1 | 15.5 | - | - | 1.27 | 1.27 | 0.06 |
| UJI 1 | 15.8 | 1.00 | 0.98 | 1.10 | 1.10 | 0.08 |
| UJI 2 | 15.3 | 1.00 | 1.02 | 1.04 | 1.08 | 0.10 |
| UJIB 1 | 2.2 | - | - | 1.09 | 1.09 | 0.21 |
| UJIB 2 | 1.8 | - | - | 1.11 | 1.11 | 0.01 |
| UTS 1 | 17.3 | - | 1.02 | 1.08 | 1.07 | 0.14 |
| Average | 13.94 | 1.00 | 1.00 | 1.28 | 1.28 | 0.08 |
| | | | **MESS** | | | |
| DSI 1 | 2.98 | - | - | 1.06 | 1.06 | 0.47 |
| LIB 1 | 6 | - | 1.00 | 0.93 | 0.93 | 0.26 |
| LIB 2 | 6 | - | 1.00 | 1.15 | 1.13 | 0.30 |
| MAN 1 | 55 | - | - | 1.03 | 1.03 | 0.02 |
| MAN 2 | 5 | - | - | 1.00 | 1.00 | 0.20 |
| MINT 1 | 13.16 | - | - | 0.90 | 0.90 | 0.07 |
| SIM 1 | 5 | - | - | 0.87 | 0.87 | 0.20 |
| TUJI 1 | 2.5 | - | - | 1.24 | 1.24 | 0.72 |
| UJI 1 | 10.65 | 1.00 | 1.00 | 1.02 | 1.06 | 2.21 |
| UJI 2 | 5.26 | 1.00 | 1.00 | 1.02 | 1.06 | 2.21 |
| UJIB 1 | 15.25 | - | - | 1.11 | 1.11 | 0.10 |

Continuation of Table B.1

| Dataset | CR | $\tilde{\zeta}_{\mathcal{B}\alpha}$ [-] | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] |
|---|---|---|---|---|---|---|
| UJIB 2 | 12 | - | - | 0.90 | 0.90 | 0.10 |
| UTS 1 | 3.11 | - | 0.98 | 1.07 | 1.07 | 3.56 |
| Average | 10.91 | 1.00 | 0.96 | 1.06 | 1.06 | 0.73 |

**PCA EWO8**

| | | | | | | |
|---|---|---|---|---|---|---|
| DSI 1 | 21.5 | - | 1 | 1.33 | 1.33 | 0.10 |
| LIB 1 | 101.5 | - | 1.00 | 0.98 | 0.99 | 0.04 |
| LIB 2 | 76.6 | - | 0.99 | 1.48 | 1.46 | 0.11 |
| MAN 1 | 13.1 | - | 1 | 1.20 | 1.20 | 0.03 |
| MAN 2 | 32.7 | - | 1 | 1.27 | 1.27 | 0.06 |
| MINT 1 | 12.8 | - | 1 | 1.15 | 1.15 | 0.05 |
| SIM 1 | 6.2 | - | 1 | 0.89 | 0.89 | 0.03 |
| TUJI 1 | 36.2 | - | 1 | 1.22 | 1.22 | 0.18 |
| UJI 1 | 36.8 | 1.00 | 0.98 | 1.05 | 1.07 | 0.15 |
| UJI 2 | 35.7 | 1.00 | 1.02 | 1.02 | 1.08 | 0.17 |
| UJIB 1 | 5.1 | - | 1 | 1.07 | 1.07 | 0.13 |
| UJIB 2 | 4.3 | - | 1 | 1.12 | 1.12 | 0.10 |
| UTS 1 | 40.4 | - | 1.04 | 1.09 | 1.09 | 0.20 |
| Average | 32.53 | 1.00 | 1.00 | 1.14 | 1.15 | 0.10 |

**AE EWO8**

| | | | | | | |
|---|---|---|---|---|---|---|
| DSI 1 | 21.5 | - | 1 | 1.30 | 1.30 | 0.12 |
| LIB 1 | 101.5 | - | 0.98 | 1.17 | 1.14 | 0.03 |
| LIB 2 | 76.6 | - | 0.98 | 1.63 | 1.60 | 0.10 |
| MAN 1 | 13.1 | - | 1 | 1.60 | 1.60 | 0.03 |

Continuation of Table B.1

| Dataset | CR | $\tilde{\zeta}_{\mathcal{B}\alpha}$ [-] | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] |
|---|---|---|---|---|---|---|
| MAN 2 | 32.7 | - | 1 | 1.59 | 1.59 | 0.15 |
| MINT 1 | 12.8 | - | 1 | 1.02 | 1.02 | 0.04 |
| SIM 1 | 6.2 | - | 1 | 1.23 | 1.23 | 0.04 |
| TUJI 1 | 36.2 | - | 1 | 1.32 | 1.32 | 0.04 |
| UJI 1 | 36.8 | 1.00 | 0.99 | 1.12 | 1.11 | 0.08 |
| UJI 2 | 35.7 | 1.00 | 1.02 | 1.07 | 1.10 | 0.10 |
| UJIB 1 | 5.1 | - | 1 | 1.07 | 1.07 | 0.15 |
| UJIB 2 | 4.3 | - | 1 | 1.09 | 1.09 | 0.07 |
| UTS 1 | 40.4 | - | 1.01 | 1.07 | 1.07 | 0.13 |
| Average | 32.53 | 1.00 | 1.00 | 1.25 | 1.25 | 0.08 |
| | | | **MESSy EWO8** | | | |
| DSI 1 | 6.9 | - | - | 1.07 | 1.07 | 0.37 |
| LIB 1 | 14.0 | - | 1.00 | 0.95 | 0.96 | 0.25 |
| LIB 2 | 14.0 | - | 1.00 | 1.29 | 1.28 | 0.33 |
| MAN 1 | 128.3 | - | - | 1.14 | 1.14 | 0.01 |
| MAN 2 | 11.7 | - | - | 1.05 | 1.05 | 0.15 |
| MINT 1 | 30.7 | - | - | 1.01 | 1.01 | 0.02 |
| SIM 1 | 11.7 | - | - | 1.02 | 1.02 | 0.06 |
| TUJI 1 | 5.8 | - | - | 1.34 | 1.34 | 0.45 |
| UJI 1 | 24.8 | 1.00 | 0.85 | 1.52 | 1.46 | 0.61 |
| UJI 2 | 12.3 | 1.00 | 0.99 | 1.03 | 1.08 | 1.59 |
| UJIB 1 | 35.6 | - | - | 1.02 | 1.02 | 0.05 |
| UJIB 2 | 28.0 | - | - | 1.23 | 1.23 | 0.11 |
| UTS 1 | 7.2 | - | 0.99 | 1.06 | 1.08 | 3.04 |

Continuation of Table B.1

| Dataset | CR | $\tilde{\zeta}_{\mathcal{B}\alpha}$ [-] | $\tilde{\zeta}_{\mathcal{F}\alpha}$ [-] | $\tilde{\varepsilon}_{3D\alpha}$ [-] | $\tilde{\varepsilon}_{2D\alpha}$ [-] | $\tilde{\tau}_{\alpha}$ [-] |
|---|---|---|---|---|---|---|
| Average | 25.47 | 1.00 | 0.97 | 1.13 | 1.13 | 0.54 |
| **Principal MESSy EWO8** | | | | | | |
| DSI 1 | 30.3 | - | - | 5.88 | 5.88 | 0.34 |
| LIB 1 | 270.7 | - | 0.97 | 1.27 | 1.21 | 0.09 |
| LIB 2 | 183.9 | - | 0.94 | 1.76 | 1.64 | 0.13 |
| MAN 1 | 399.3 | - | - | 2.40 | 2.40 | 0.01 |
| MAN 2 | 163.3 | - | - | 2.71 | 2.71 | 0.06 |
| MINT 1 | 67.5 | - | - | 0.96 | 0.96 | 0.02 |
| SIM 1 | 18.7 | - | - | 1.01 | 1.01 | 0.05 |
| TUJI 1 | 50.2 | - | - | 1.87 | 1.87 | 0.38 |
| UJI 1 | 201.8 | 1.00 | 0.77 | 1.79 | 1.61 | 0.12 |
| UJI 2 | 95.2 | 1.00 | 0.80 | 1.36 | 1.38 | 0.26 |
| UJIB 1 | 53.4 | - | - | 1.34 | 1.34 | 0.03 |
| UJIB 2 | 36.2 | - | - | 1.62 | 1.62 | 0.03 |
| UTS 1 | 51.4 | - | 0.93 | 1.35 | 1.23 | 0.44 |
| Average | 124.76 | 1.00 | 0.88 | 1.95 | 1.91 | 0.15 |

End of Table B.1