DGK Ausschuss Geodäsie (DGK)
der Bayerischen Akademie der Wissenschaften

Reihe C Dissertationen Heft Nr. 955

Duo Wang

Deep Learning Approaches for GNSS and InSAR Geodetic Modeling

München 2025

Verlag der Bayerischen Akademie der Wissenschaften, München

ISSN 0065-5325 ISBN 978-3-7696-5367-0

DGK Ausschuss Geodäsie (DGK)
der Bayerischen Akademie der Wissenschaften

Reihe C Dissertationen Heft Nr. 955

Deep Learning Approaches for GNSS and InSAR Geodetic Modeling

Zur Erlangung des akademischen Grades eines Doktors der Ingenieurwissenschaften (Dr.-Ing.)
von der Fakultät für Bauingenieur-, Geo- und Umweltwissenschaften
des Karlsruher Instituts für Technologie (KIT)
genehmigte Dissertation

von

M.Sc. Duo Wang

Geb. in Beijing

München 2025

Verlag der Bayerischen Akademie der Wissenschaften, München

ISSN 0065-5325 ISBN 978-3-7696-5367-0

Adresse des Ausschusses Geodäsie (DGK)
der Bayerischen Akademie der Wissenschaften:

Ausschuss Geodäsie (DGK) der Bayerischen Akademie der Wissenschaften

Alfons-Goppel-Straße 11 ● D – 80 539 München

Telefon +49 – 89 – 23 031 1113 ● Telefax +49 – 89 – 23 031 - 1283 / - 1100
e-mail post@dgk.badw.de ● http://www.dgk.badw.de

Referent:	Prof. Dr.-Ing. Hansjörg Kutterer
Korreferentin:	Prof. Dr. Benedikt Soja

Tag der mündlichen Prüfung: 26.02.2025

Diese Dissertation ist auf dem Server des Ausschusses Geodäsie (DGK)
der Bayerischen Akademie der Wissenschaften unter <http://dgk.badw.de/>
sowie auf dem Server des Karlsruher Instituts für Technologie (KIT) unter
<https://doi.org/10.5445/IR/1000181333> elektronisch publiziert

ISSN 0065-5325

ISBN 978-3-7696-5367-0

Abstract

Global Navigation Satellite Systems (GNSS) and Interferometric Synthetic Aperture Radar (InSAR) are crucial geodetic technologies that utilize satellite data to precisely monitor the Earth's surface. Both methods rely on microwave signals emitted by satellites, enabling the detection of activities on the Earth’s surface and within the atmosphere. However, when processing these signals, it is inevitable to introduce external models to extract the parts of the signals that are effective for geodetic observation. Therefore, the accuracy and efficiency of modeling have become important factors that directly affect these geodetic methods, and have also become central concerns within the geodetic research community.

Traditionally, modeling of GNSS and InSAR has depended on external data, such as Numerical Weather Models (NWM), or statistical and filtering approaches. These traditional methods, however, often suffer from limitations in terms of accuracy, spatial resolution, and computational efficiency. Deep learning, a data-driven approach within artificial intelligence, offers advanced nonlinear modeling capabilities and has shown significant promise in handling spatio-temporal geodata. This thesis focuses on two key applications of deep learning for geodesy: Zenith Tropospheric Delay (ZTD) time series modeling and Distributed Scatterer (DS) identification and prediction.

For tropospheric delay modeling, this study introduces a novel deep learning framework, called Gaussian Mixed Long Short-Term Memory Network (GM-LSTM), which utilizes Bi-LSTM to capture the Zenith Wet Delay (ZWD) patterns from GNSS observations and correct ZWD estimates produced by NWM ray tracing. A key contribution of this research is the development of the first deep learning-based ZTD estimation algorithm that integrates geodetic background knowledge. Compared to traditional methods, such as ERA5 ray tracing, VMF3, and GACOS—known for centimeter-level errors—and general deep learning methods like DNN lacking geodetic context, the GM-LSTM framework was validated across eight distinct latitude regions in Europe, achieving an average RMSE of 4.6 mm. This establishes GM-LSTM as a state-of-the-art solution for ZTD modeling. Furthermore, GM-LSTM introduces the use of probability density to describe ZWD, effectively accounting for spatial uncertainties. Meteorological data confirm that GM-LSTM can accurately reflect uncertainties due to spatially variable rainfall events. When trained on homogeneous data, the model excels in predicting ZTD during heavy rainfall, outperforming other methods.

In addressing DS identification and prediction, this thesis presents the Distributed Scatterer Prediction Network (DSPN), a convolutional neural network designed to predict DS candidates from a pair of polarimetric SAR images. The primary contribution of DSPN lies in significantly accelerating the traditional InSAR deformation analysis without sacrificing accuracy. This research demonstrates, for the first time, that deep learning can accurately identify DS candidates at minimal computational cost before time-consuming preprocessing steps, thereby reducing overall computational demands and enhancing the efficiency of InSAR workflows.

This study highlights the potential of deep learning in geodesy by addressing two key modeling challenges and emphasizes the importance of combining geodetic domain knowledge with deep learning frameworks. This combination not only enhances the interpretability of the model but also improves performance, and is expected to open up new approaches to geodetic modeling practice.

Zusammenfassung

Globale Satellitennavigationssysteme (GNSS) und Interferometrisches Radar mit Synthetischer Apertur (InSAR) sind wichtige geodätische Technologien, die Satellitendaten nutzen, um die Erdoberfläche präzise zu überwachen. Beide Methoden basieren auf von Satelliten ausgesandten Mikrowellensignalen, die die Erfassung von Aktivitäten auf der Erdoberfläche und innerhalb der Atmosphäre ermöglichen. Bei der Verarbeitung dieser Signale ist es jedoch unvermeidlich, externe Modelle einzuführen, um die für geodätische Beobachtungen relevanten Signalanteile zu extrahieren. Somit sind die Genauigkeit und Effizienz der Modellierung zu wichtigen Faktoren geworden, die sich auf diese geodätischen Methoden unmittelbar auswirken, und stellen auch zentrale Gegenstände innerhalb der geodätischen Forschungsgemeinschaft dar.

Herkömmlich basierte die Modellierung in GNSS und InSAR auf externen Daten, wie denen von numerischen Wettermodellen (NWM), oder auf statistischen und Filteransätzen. Diese herkömmlichen Methoden haben jedoch oft Unzulänglichkeiten in Bezug auf Genauigkeit, räumliche Auflösung und Recheneffizienz. Deep Learning, ein datengetriebener Ansatz aus dem Bereich der Künstlichen Intelligenz, bietet die Möglichkeit einer fortschrittlichen. nichtlinearen Modellierung und hat sich als vielversprechend in der Verarbeitung raumzeitlicher Geodaten erwiesen. Die vorliegende Arbeit konzentriert sich auf zwei Schlüsselanwendungen von Deep Learning in der Geodäsie: die Modellierung der Zenith Tropospheric Delay (ZTD) und die Identifikation und Vorhersage von Distributed Scatterern (DS).

Für die Modellierung der troposphärischen Verzögerung führt diese Studie ein neuartiges Deep-Learning-Framework ein, genannt Gaussian Mixed Long Short-Term Memory Network (GM-LSTM), das Bi-LSTM nutzt, um Muster des Zenith Wet Delay (ZWD) aus GNSS-Beobachtungen zu erfassen und ZWD-Schätzungen zu korrigieren, die durch Strahlverfolgung aus Daten von NWM erzeugt wurden. Ein wesentlicher Beitrag dieser Forschung ist die Entwicklung des ersten auf Deep Learning basierenden ZTD-Schätzalgorithmus, der geodätisches Kontextwissen integriert. Im Vergleich zu traditionellen Methoden, wie der Strahlverfolgung basierend auf ERA5, VMF3 und GACOS — die bekannterweise Fehler im Zentimeterbereich aufweisen — und allgemeinen Deep-Learning-Methoden wie DNN, die kein geodätisches Wissen einfließen lassen, wurde das GM-LSTM-Framework für acht verschiedene Breitengrade in Europa validiert und erzielte dabei einen durchschnittlichen RMSE von 4,6 mm. Dies erweist GM-LSTM als eine hochgenaue Lösung für die ZTD-Modellierung. Methodisch führt GM-LSTM die Nutzung von Wahrscheinlichkeitsdichten bei der Schätzung von ZWD ein, was es erlaubt, räumliche Unsicherheiten effektiv zu berücksichtigen. Meteorologische Daten bestätigen, dass GM-LSTM Unsicherheiten aufgrund räumlich variabler Regenereignisse genau widerspiegeln kann. Bei homogener Datenbasis übertrifft das Modell andere Methoden bei der Vorhersage von ZTD während starker Regenfälle.

Im Bereich der DS-Identifikation und -Vorhersage stellt diese Arbeit das Distributed Scatterer Prediction Network (DSPN) vor, ein Convolutional Neural Network, das entwickelt wurde, um DS-Kandidaten aus einem Paar polarimetrischer SAR-Bilder vorherzusagen. Der Hauptbeitrag von DSPN liegt in der signifikanten Beschleunigung der traditionellen InSAR-Deformationsanalyse, ohne dabei an Genauigkeit einzubüßen. Diese Forschung zeigt zum

ersten Mal, dass Deep Learning DS-Kandidaten mit minimalem Rechenaufwand genau identifizieren kann, bevor zeitaufwändige Vorverarbeitungsschritte durchgeführt werden. Dadurch werden die Gesamtrechenanforderungen reduziert und die Effizienz der InSAR-Arbeitsabläufe verbessert.

Die vorliegende Studie beleuchtet das Potenzial von Deep Learning in der Geodäsie, indem sie zwei zentrale Herausforderungen der Modellierung behandelt und die Bedeutung der Kombination von geodätischem Fachwissen mit Deep Learning-Rahmenwerken herausstellt. Die Kombination verbessert nicht nur die Interpretierbarkeit der Modelle, sondern auch ihre Leistung und dürfte neue Ansätze für die Praxis der geodätischen Modellierung eröffnen.

Contents

Abbreviations

Acc	Accuracy
ACF	Auto Correlation Function
Adam	Adaptive moment estimation
ANN	Artificial Neural Networks
BFGS	Broyden–Fletcher–Goldfarb–Shanno algorithm
BP	Backpropagation
BPTT	Backpropagation through time
cdf	Cumulative distribution function
CNN	Convolutional neural networks
DEM	Digital Elevation Models
DNN	Deep artificial neural networks
DORIS	Doppler Orbiting and Radio positioning Integrated by Satellite
DS	Distributed Scatterers
DSC	Distributed Scatterers Candidates
DSI	Distributed Scatterer Interferometry
DSPN	Distributed Scatterer Prediction Network
ECMWF	European Center for Medium-Range Weather Forecasts
ERA5	the fifth generation of European Reanalysis
FN	False negatives
FNR	False Negative Rate
FP	False positives
GACOS	Generic Atmospheric Correction Online Service for InSAR
GM-LSTM	Gaussian Mixture Long Short-Term Memory Network
GMM	Gaussian mixture model
GNSS	Global Navigation Satellite Systems
GNSS-PPP	GNSS Precise Point Positioning
GPT3	Global Pressure and Temperature model 3
GPU	Graphics processing unit
IDW	Inverse Distance Weighting interpolation
InSAR	Interferometric Synthetic Aperture Radar
IPP	Ionospheric pierce point
ITD	Iterative Tropospheric Decomposition
IWV	Integrated Water Vapor
KS test	Kolmogorov-Smirnov test
LC	Lost coverage
LCC	Land cover classification
LOS	Line-of-sight
LSTM	Long Short-Term Memory Network
MAPE	Mean Absolute Percentage Error
MBE	Mean Bias Error
ML	Maximum likelihood
MLE	Maximum Likelihood Estimation
MLP	Multi-layer perceptrons
MODIS	Moderate Resolution Imaging Spectroradiometer
MT-InSAR	Multi-Temporal InSAR
NGL	Nevada Geodetic Laboratory
NPV	Negative Predictive Value
NWM	Numerical Weather Models
pdf	Probability density function
PELU	Piecewise Exponential Linear Unit
Pre	Precision
PS	Persistent Scatterers
PSC	PS Candidates
PSI	Persistent Scatterer Interferometry
PTA	Phase Triangulation Algorithm
Rec	Recall
ReLU	Rectified Linear Unit
RMSE	Root Mean Square Error
RMSProp	Root Mean Square Propagation
RNN	Recurrent neural networks

SAR	Synthetic Aperture Radar
SBAS	Small Baseline Subset
SE	Standard Error
SGD	Stochastic Gradient Descent
SHP	Statistical Homogeneous Pixels
SLC	Single Look Complex
SLM	Single-Layer Model for the ionosphere
SNR	Signal-to-noise ratio
StaMPS	Stanford Method for Persistent Scatterers
STEC	Slant Total Electron Content
SVD	Singular Value Decomposition
tanh	Hyperbolic tangent
TEC	Total Electron Content
TN	True negatives
TP	True positives
VLBI	Very Long Baseline Interferometry
VMF3	Vienna Mapping Functions 3
ZHD	Zenith Hydrostatic Delay
ZTD	Zenith Total Delay
ZWD	Zenith Wet Delay

Symbols

B_L	Integer carrier phase ambiguity and hardware biases of GNSS signals
φ_L	Received GNSS signals of carrier phase
$*^H$	Hermitian conjugation
$\Delta\phi_n$	Wrapped residual phase
$\Delta\varphi_{topo}$	DEM error
$\Delta\varphi_{topo}^{sc}$	Spatially correlated part of DEM error
$\Delta\varphi_{topo}^{su}$	Spatially uncorrelated topographic error
p_{w_i}	Water vapor pressure at i-th layer pressure level
SD_0	Stratified component delay at sea level
ΔD	Displacement in LOS direction
Δs^{ion}	Ionospheric delay
Δs_h^{trop}	Hydrostatic delay
Δs_w^{trop}	Wet delay
$\Delta\theta^{su}$	Spatially uncorrelated look angle error
$\Delta\phi$	Interferometric phase
∇	Gradient
$\angle$	Argument of a complex number
$\odot$	Hadamard product
A	Amplitude
b	Bias of neural network
B	Baseline of two acquisitions
b^S	Frequency-dependent hardware delays of the satellite
$B_{\parallel}$	Parallel baseline
$B_{\perp}$	Perpendicular baseline
B_0	Ambient magnetic field strength
b_R	Frequency-dependent hardware delays of the receiver
C	Adjacency matrix for SBAS connection graph
C_{DS}	Number of DS in a cell
C_m	Number of DSC predicted by DSPN mask in a cell
D	Distance between the satellite and the surface target
$d_{i,i+1}$	Geometric height difference between the altitude of i-th and i+1-th pressure level
D_{KL}	Kullback-Leibler divergence
D_N	Absolute difference between two cdf
DOY	Day of year
e	Electron charge
E_{GT}	Expectation of the distribution from the ground truth
f	Wave frequency
F(z)	Mapping function for ionospheric delay
g	Earth's gravitational acceleration
G_i	Geopotential at i-th layer pressure level
GT	Ground truth
H	Altitude of SLM layer
h^t	Hidden state of RNN at time t
H_{CE}	cross-entropy
h_s	surface altitude
$h_t^{\rightarrow}$	Forward hidden state of Bi-LSTM
$h_t^{\leftarrow}$	Backward hidden state of Bi-LSTM
h_{top}	Height of tropopause
H_{WCE}	Weighted cross-entropy
HOD	Hour of day
IWV_{ERA5}	IWV retrieved from ERA5
IWV_{GPT3}	IWV retrieved from GPT3
L_{DNN}	Likelihood of the output distribution of the DNN
LAT	Latitude
LON	Longitude
lr	Learning rate

M	Master acquisition
M_d	Molar mass of dry air
m_e	Electron mass
$mf_h(\varepsilon)$	Mapping function for hydrostatic delay
$mf_w(\varepsilon)$	Mapping function for wet delay
N	Refractivity of troposphere
N_e	Electron density
n_{ion}	Refractive index of ionosphere
n_{trop}	Refractive index of troposphere
o^t	Output of RNN at time t
P	Received pseudorange code
p_d	Pressure of dry atmosphere
p_w	Pressure of wet atmosphere
R	Mean Earth radius
r	Random error
R^2	Coefficient of determination
RH_i	Relative humidity at i-th layer pressure level
S	Slave acquisition
s	Geometric distance between receiver and satellite
s^{ion}	Propagation distance of microwaves
S^{ion}	Actual path of the signal through the ionosphere
S^{trop}	Actual path of signal through troposphere
s_0^{ion}	Geometric distance of microwave propogation
S_0^{ion}	Straight path of the signal through the ionosphere
S_0^{trop}	Straight path of signal through troposphere
SD	Altitude-related stratified delay
T	Temperature
T_a	Actual running time
T_i	Temperature at i-th layer obtained from ERA5
T_m	Weighted mean temperature
T_t	Total computation time
TD	Elevation-related turbulent delay
W	Wrapping operator
W^{-1}	Unwrapping operator
w_i	Weight of neural network
z	SAR signal
Z_d	Compressibility factors for dry atmosphere
Z_W	Compressibility factors for wet atmosphere
ZHD_{VMF3}	ZHD retrieved by VMF3
ZTD_{GNSS}	ZTD retrieved by GNSS
ZWD_{ERA5}	ZWD retrieved by ERA5 ray tracing
ZWD_{GNSS}	ZWD retrieved by GNSS
ZWD_{VMF3}	ZWD retrieved by VMF3
α	Mixing coefficients
β	Coefficient factor
$\hat{\Gamma}$	Estimated coherence matrix
γ	Coherence
γ_{PTA}	Coherence estimated by PTA
Δs^{trop}	Slant tropospheric delay
δt^S	Receiver and satellite clock offsets to the GNSS system time
δt_R	Receiver clock offsets to the GNSS system time
ε	Elevation angle
ϵ_0	Permittivity of free space
θ	Angle between the ambient magnetic field vector and the wave vector
θ_0	SAR look angle
θ_{inc}	SAR incidence angle
κ	Curvature effect
λ	Wavelength
μ	Mean
ρ	Density of air
σ	Activation function
σ^2	Variance
ϕ	Phase shift between the emitted and

	received signals
ϕ^{sc}	Spatially correlated phase
φ_{atmo}	Atmospheric phase difference
ϕ_{atmo}	Phase delay due to atmospheric effects
φ_{defo}	Deformation phase
φ_{flat}	Flat earth phase
φ_{h}	Topography phase
φ_{noise}	Noise phase
φ_{orbit}	Orbit error phase
ϕ_{scat}	Phase change caused by the scattering characteristics
ω	Angular frequency
ω_0	Electron plasma frequency
ω_H	Electron gyro frequency

1. Introduction

1.1. Motivation

High-precision and large-scale Earth observation is the cornerstone of geodetic science, especially in the fields of spaceborne geodesy and remote sensing. Comprehensive analysis and interpretation of signals acquired by satellite sensors, coupled with data from a global observing network, can provide insights into earth surface and atmospheric changes. This understanding holds paramount significance within environmental and earth sciences as well as engineering disciplines. Its irreplaceable role extends to the applications of geohazard assessment, engineering measurements, atmospheric monitoring, and other fields. The Earth's surface undergoes perpetual motion, driven by natural phenomena such as earthquakes, plate tectonics, and volcanic activity, as well as anthropogenic activities, including mineral extraction, oil drilling, groundwater depletion, urban development, and civil engineering projects (Elliott et al., 2016; Seo et al., 2023). These motions often yield substantial impacts, predominantly adverse, on human society and the environment alike. Atmospheric monitoring is also critical to protecting human society from threats posed by phenomena such as global warming and extreme climate disasters. Water in the atmosphere has become a key factor that profoundly affects human survival. Its movements are not only closely related to the occurrence of floods or droughts caused by excessive or insufficient rainfall but also have a significant impact on the entire climate system. Given the intricate interactions within the water vapor-cloud-rain cycle, detecting water vapor is of critical importance for improving understanding of global warming and climate change dynamics, as it directly affects the redistribution of energy in the Earth's atmosphere (Worden et al., 2007; Trenberth et al., 2003).

Despite their distinct original purposes, two disparate spaceborne microwave remote sensing technologies, namely Global Navigation Satellite Systems (GNSS) and Interferometric Synthetic Aperture Radar (InSAR), serve as invaluable geodetic instruments for earth observation. Initially focused on precise positioning and navigation applications, GNSS technology has evolved significantly over time. With the establishment of Continuously Operating Reference Station (CORS) GNSS networks and advancements in GNSS technology, it has emerged as an indispensable tool for deformation measurement at monitoring stations. As a continuously operating precise earth observation technology, it can accurately monitor subtle changes in the earth's surface over time (Teunissen & Montenbruck, 2017). With the help of high-precision timing systems, GNSS satellites send encoded orbit and time signals through electromagnetic waves that propagate at the speed of light. After the signal is captured by the ground receiver, it can use the timing system to calculate the time it takes for the signal to propagate from the satellite antenna to the receiver antenna. This time multiplied by the speed of light can be used to know the distance of the satellite from the receiver. Since the orbit information of the satellite can be considered to be accurately obtained, when the receiver receives more than three signals transmitted by satellites at different locations at the same time, its three-dimensional position Information can be determined. For the CORS receiver, it can resolve position information every 15 seconds, thereby producing an ultra-high time resolution surface deformation time series.

Indeed, electromagnetic waves encounter refraction as they propagate through the atmosphere, deviating from the straight-line distance observed in space between the satellite and the receiver. This atmospheric refraction introduces delays in the propagation of signals. By analyzing these delayed signals, it becomes possible to derive valuable information regarding the water vapor content and variable electron content present in the atmosphere above the receiver. This capability underscores the versatility of GNSS technology, extending its utility beyond precise positioning and navigation applications to include atmospheric monitoring and meteorological studies. However, GNSS can only provide precise point observations of atmospheric activities above a station but cannot provide spatially continuous observation products, which limits its application in the meteorological field. Although the CORS network can provide station-based GNSS measurement products spaced tens of kilometers apart, their spatial resolution remains inadequate for fulfilling the requirements of large-scale surface measurements. This means that using GNSS to observe regional tropospheric delays and water vapor activity remains an open question for the geodetic community.

As a complementary technology operating on the same physical principles, InSAR is indeed a 'Radio Detection And Ranging' technology capable of observing large-scale surface deformation with remarkable millimeter-level accuracy (Ferretti et al., 2007). Unlike GNSS technology, InSAR operates by detecting changes in the sensor-to-ground distance through analyzing the phase difference between Synthetic Aperture Radar (SAR) acquisitions of the same location at different times. Since SAR operates by emitting its own microwaves and receiving their echoes, it allows InSAR technology to overcome limitations associated with receivers. This capability enables spaceborne SAR systems to conduct large-scale mapping at high altitudes in a single pass. For instance, the interferometric wide-swath acquisition mode of the Sentinel-1 SAR satellite can capture a swath of 250 km with a spatial resolution of 5×20 meters in a single image.

However, the reliance on satellite revisits means that two SAR acquisitions are typically separated by several days. This not only constrains the temporal resolution of Earth observations but also introduces decorrelation issues stemming from atmospheric changes or alterations in ground objects. These factors can ultimately lead to solution failure in InSAR analysis (Hooper et al., 2007; Michaelides et al., 2019). To address this limitation and analyze the long-term ground deformation, researchers have developed various Multi-Temporal InSAR (MT-InSAR) algorithms aimed at mitigating the impact of decorrelation. One such technology is Persistent Scatterer Interferometry (PSI) (Ferretti et al., 2007; Hooper et al., 2007), a subtype of MT-InSAR. This algorithm has proven successful in monitoring the deformation of artificial structures by selectively identifying and processing scatterers with stable and dominant scattering properties. For natural targets such as wilderness, forests, farmland, bare soil, and rock surfaces, it's often challenging for them to exhibit a dominant signal in the scatterer cell. However, if a sufficiently large group of adjacent pixels can be identified sharing the same scattering mechanism, statistical estimation can be employed to assess the portion of noise responsible for decorrelation. From these properties, Distributed Scatterer Interferometry (DSI) technology has been developed (Even & Schulz, 2018).

DSI primarily encompasses two distinct technical routes: The first approach involves utilizing SAR acquisition pairs that are closely spaced in both time and space to construct interferogram subsets. This technique is also known as Small Baseline Subset (SBAS) technology (Berardino et al., 2002). By leveraging redundant sets of small baseline interferograms, SBAS mitigates the effects of decorrelation while enhancing estimation robustness through redundancy. The second method involves leveraging all possible pairs of interferograms to reduce the random noise associated with Distributed Scatterers (DS). Subsequently, the derived DS can be treated similarly to Persistent Scatterers (PS) and jointly processed (Gaddes et al., 2019; Jiang et al., 2015; Hooper, 2008; Ferretti et al., 2011). By jointly processing both PS and DS, a larger and denser set of scatterers can be analyzed, leading to a more robust estimation of the surface deformation field. While DSI offers the advantage of increasing the number of analyzable scatterers, the computational cost associated with selecting these Distributed Scatterers Candidates (DSC) is substantial, resulting in slow processing speeds. The computational bottleneck of DS processing primarily occurs during its preprocessing stage, specifically in selecting DSC and estimating their phase triangulation coherence. This estimation necessitates complex nonlinear calculations, causing the DS preprocessing step to take several times longer than PS processing. For instance, in the case of Sentinel-1 acquisitions covering approximately 420 km^2 around the south side of volcano Etna in Sicily, Italy, even when utilizing advanced multi-threaded parallel computing technology, DS preprocessing can take approximately 446 hours in an 8-thread working environment (Wang et al., 2022). Fortunately, certain ground covers, such as water, forests, and shadows, are not suitable candidates for DS analysis. Therefore, implementing appropriate pattern recognition techniques to discard these inappropriate points before DS preprocessing is expected to reduce DS preprocessing time and improve the overall processing efficiency, which makes identifying DS patterns another open question of concern to the geodetic community.

In recent years, the advancement of GPU parallel computing technology has catalyzed the evolution of machine learning and artificial intelligence methodologies, furnishing powerful tools for addressing pattern recognition challenges. Consequently, significant strides have been made in domains such as computer vision and natural language processing (Goodfellow et al., 2016; Lecun et al., 2015). In optical and multispectral remote sensing, where tasks like ground object classification and recognition bear resemblances to image classification and semantic segmentation in computer vision, deep learning techniques have been effectively leveraged (Ma et al., 2019; Zhu et al., 2017). These methodologies have enabled the successful classification of ground objects, detection of vegetation, and identification of water bodies, thereby heralding breakthroughs in remote sensing applications. Recently, the geodetic community has increasingly recognized the potential of data-driven deep learning methods in modeling and has begun exploring their application in geodesy, including GNSS, gravity field mass changes, Earth orientation parameters, and deformation monitoring. For example, Tang et al. (2024) and Yang & Fang (2023) used Autoformer deep learning with multilayer perceptron and generative adversarial network to predict the global Total Electron Content (TEC) in the ionosphere respectively, Shi et al. (2023) used dense neural network to generate CORS-based Zenith Total Delay (ZTD) dataset, Gou & Soja (2024) used convolutional neural networks convolutional neural networks to assimilate Total water storage anomalies with the Gravity Recovery and Climate Experiment (GRACE) and its follow-on GRACE-FO satellite data, Shahvandi et al. (2023) proposed a machine learning algorithm named ResLearner to improve the accuracy of rapid and predicted Earth orientation parameters. These works showcase the potential of deep learning in this domain.

However, geodesy, with its focus on the precise measurement of the Earth, involves highly specialized data in both geodetic models and their applications. This is the primary distinction between geodetic tasks and computer vision tasks in remote sensing. For instance, in remote sensing, the 'Ground Truth' for object recognition can be easily obtained through manual labeling. In contrast, geodetic applications, such as delay estimation in GNSS solutions or DSC detection in InSAR processing, are highly dependent on domain-specific geodetic knowledge. This reliance implies that there are no standardized datasets or definitive answers for model training, which poses a significant challenge in adapting deep learning methods to meet the specific needs of geodesy. In addition, the application of AI methods in the geodetic community tends to rely on existing computer science algorithms, treating them as regressors or classifiers for geodetic tasks, often without fully integrating geodetic expertise. This lack of synergy between deep learning algorithms and the professional knowledge of geodesy has limited the further development of deep learning in geodetic applications.

Addressing the above two open questions respectively, this work proposes two new deep learning frameworks specifically designed to integrate geodetic background knowledge for those two tasks: GNSS tropospheric delay modeling and DSC prediction. Compared to traditional non-deep learning algorithms, the two proposed deep learning methods adopt data-driven approaches, allowing them to adaptively capture the spatiotemporal patterns in GNSS and InSAR data. As a result, they achieve state-of-the-art performance in both tasks. More importantly, this study does not simply apply existing deep learning algorithms to predict geodetic parameters but integrates geodetic background knowledge into deep learning algorithms. Experimental results show that the proposed algorithms with integrating geodetic background knowledge in both tasks exhibit significant advantages over the standard deep learning model. More importantly, this study goes beyond the mere application of existing deep learning algorithms for predicting geodetic parameters. It incorporates geodetic background knowledge directly into the deep learning framework. The experimental results demonstrate that the proposed algorithm, supplemented with geodetic context, outperforms standard deep learning models across both tasks. From a geodetic modeling perspective, the two proposed methods offer deep learning frameworks tailored to different dimensions—space and time. Given the adaptability of the algorithm, it also holds significant reference value for related tasks such as SAR image feature extraction, InSAR atmospheric correction, and 3D water vapor field construction.

1.2. Scientific Objectives

In this thesis, the main scientific objective discussed is how to use deep learning to solve modeling challenges in geodetic applications, which mainly focuses on the integration of specialized geodetic knowledge and deep learning algorithms. Although some researchers have begun using deep learning methods in geodesy, e.g. Yang et al. (2021), Zhang et al. (2024), Klos et al. (2023), they commonly regard these deep learning models as advanced alternatives to traditional regressors or classifiers, using existing algorithms in the field of artificial intelligence to process geodetic data. While those approaches may gain the benefits offered by the data-driven property of the deep learning approach compared to traditional algorithms, the lack of domain-specific knowledge greatly limits the performance of general artificial intelligence algorithms. Unlike "standard problems" in computer science, such as image classification and segmentation or time series analysis, deep learning for geodesy is characterized by lacking well-defined problem specifications and standardized datasets. In computer science, researchers typically have access to well-defined problems and uniform datasets for algorithm training, which simplifies the development of solutions. In contrast, for geodetic modeling, there is usually no clear problem specification and fixed data set for training, and the process of making supervised data may significantly impact model performance and become one of the key parts of the solution. Furthermore, generic loss functions may not provide the necessary constraints to guide neural networks in learning specialized geodetic background knowledge for accurate modeling. This poses unique challenges for the geodetic community in adopting deep learning methods.

In this thesis, two geodetic modeling problems – tropospheric delay modeling and DS identification and prediction – are used to explore the deep learning modeling method that incorporates geodetic background knowledge. The main contributions of these two studies are summarized as follows:

1. Tropospheric Delay Modeling

This study introduces a deep learning time series modeling approach called Gaussian Mixture

Long Short-Term Memory Network (GM-LSTM) to extract tropospheric delay correction patterns from GNSS time series data. Different from traditional methods such as ray tracing or surface empirical models using Numerical Weather Models (NWM) and standard deep learning methods such as deep artificial neural networks (DNN), this work pioneered the use of the proposed GM-LSTM to model tropospheric delay as a probability density function which can measure the uncertainties caused by spatial heterogeneity. Compared to the centimeter-level error of traditional methods, this study achieved an average RMSE of 4.6 mm in the ZTD at eight different latitude areas in Europe. This level of accuracy represents a state-of-the-art advancement and offers important implications for applications such as InSAR atmospheric correction and water vapor field retrieval. Validation on meteorological data shows that the GM-LSTM model accurately reflects the uncertainty introduced by spatially varying rainfall and exhibits robust performance even under heavy rainfall conditions—a situation that traditional ZTD estimation techniques often struggle to address.

2. Distributed Scatterer Identification and Prediction

To address the challenge of DS identification and prediction, this thesis proposes a deep learning spatial modeling approach called Distributed Scatterer Prediction Network (DSPN). This method uses a convolutional neural network to learn the spatial pattern of DS, enabling the generation of high-precision DSC masks from a pair of polarimetric SAR images. The contribution of this work is that it applies deep learning in a novel way to predict DSC before preprocessing, thus addressing the time-consuming preprocessing step in DS-InSAR deformation analysis. More importantly, DSC, as a key intermediate product in the DS-InSAR processing flow, depends on the data and processing strategy. Therefore, it is not suitable for standard deep learning segmentation models. Although the problem can be specified to pixel-by-pixel regression using existing convolutional neural networks with phase triangulation quality numbers as labels, it may overlook critical weak DSC, such as motorway pixels. To address this, this study introduces a unique DS pseudo-label generation strategy, as well as a custom network architecture and loss function that incorporates background knowledge of InSAR processing to learn the spatial features of DSC. Evaluations of six datasets of different topographic types from North Rhine-Westphalia and Sicily showed that the proposed DSPN method has almost no DS coverage loss and saved nearly 47% of computing time without sacrificing analysis accuracy. Therefore, this deep learning method has great promise for improving the efficiency of DS-InSAR processing.

1.3. Thesis Outline

This thesis is structured into seven chapters:

Chapter 2 introduces the fundamentals of atmospheric effects in space geodesy, which emphasizes the measurement and estimation of tropospheric delay and its derived water vapor retrieval in GNSS meteorology. In this chapter, conventional methods of ZTD estimation and their limitations are introduced to illustrate the significance and value of developing a new tropospheric delay model.

Chapter 3 covers the fundamental concepts of MT-InSAR, focusing on PS and DS processing methods based on the StaMPS approach. This chapter explains the advantages and disadvantages of combined PS and DS processing and explains the significance of pre-screening DS for processing acceleration.

Chapter 4 describes the basic framework of deep learning, including the execution of feedforward neural networks and recurrent neural networks. The backpropagation (BP) algorithm used for learning and common optimizers are also introduced. This chapter doesn’t focus on specific network structures but focuses on introducing the necessary basic background.

Chapter 5 presents the GM-LSTM, a deep learning-based model for tropospheric delay estimation. This chapter details the network architecture, training methodology, and the model's range limitations. The ZTD values generated by GM-LSTM are compared with those from existing methods across eight distinct latitudes and topographic regions in Europe. The chapter also provides a comprehensive analysis of the model's performance and applicability under extreme meteorological conditions. Additionally, it introduces a deep learning approach for generating Integrated Water Vapor (IWV) in the absence of meteorological observations and high-precision NWM, with test results from four different GNSS stations in Europe.

Chapter 6 introduces the deep learning-based DS preprocessing acceleration algorithm DSPN proposed in this thesis, outlining the algorithm's workflow and training methodology. The method is tested on six distinct terrains in North Rhine-Westphalia and Sicily using Sentinel-1 data to demonstrate its effectiveness. A comparison with the standard U-Net neural network highlights the necessity of integrating deep learning algorithms with geodetic expertise. This chapter provides a detailed discussion of the method's acceleration performance and examines potential error sources to assess the reliability of the approach.

In Chapter 7, the conclusions of this thesis are summarized, and the outlook is followed for future work.

Portions of this thesis are derived from publications published or submitted by the author.

2. The fundamentals of atmospheric effects in space geodesy

This chapter introduces the basics of atmospheric effects relevant to space geodesy. First, the basic structure and composition of the atmosphere are reviewed, followed by an overview of the effects of the ionosphere and neutral atmosphere on microwave signals (also known as delays). Then, empirical models for these delays are introduced. Finally, methods for measuring atmospheric effects based on GNSS meteorology will be introduced, mainly including the measurement of the total tropospheric zenith delay and integrated water vapor inversion. Since this chapter emphasizes atmospheric effects and measurements, only relevant content related to GNSS meteorological measurements is introduced. For comprehensive literature on GNSS applications in geodesy, navigation, and positioning, please refer to the following handbook: *Springer Handbook of Global Navigation Satellite Systems* (Teunissen & Montenbruck, 2017).

2.1. Atmospheric structure

To describe the interaction between atmospheric effects and microwave signals, a basic introduction to the structure of the Earth's atmosphere has been provided first. The atmospheric composition differs significantly at different altitudes above the Earth's surface due to the gravitational and rotational forces exerted by the Earth. These large-scale atmospheric features also play a crucial role in shaping geodetic properties. Therefore, a comprehensive grasp of atmospheric structure and circulation within key layers is essential for accurately assessing the geodetic impact at different altitudes.

The Earth's atmosphere can be divided into two main layers, based on altitude and electrical properties: the ionosphere and the neutrosphere. The ionosphere is the uppermost layer, extending from about 60 to 2000 km, with particles concentrated mainly between 300 and 400 km (Bowhill, 1971; Hargreaves, 2003). It consists of electrically charged atoms or molecules (ions). In contrast, the neutrosphere, consisting of the troposphere, stratosphere, mesosphere, and part of the thermosphere, extends from the Earth's surface to the bottom of the ionosphere and is characterized by an electrically neutral state.

The troposphere is the lowest layer of the Earth’s atmosphere and contains the largest air mass. Its vertical extent typically reaches up to approximately 10 km from the surface (Schindelegger et al., 2013), although these boundaries exhibit latitude and seasonal variations. Around 80% of the total atmospheric mass is in this layer. In the troposphere, the temperature decreases with increasing altitude, and a significant portion of the Earth's water vapor is located in this layer. Consequently, the troposphere is inherently unstable, serving as the primary arena for various meteorological phenomena. Water vapor undergoes complex processes within the troposphere. When vapor pressure at a particular location within the troposphere reaches saturation (also known as saturated vapor pressure), water vapor condenses into tiny water droplets. These droplets then attach to suspended aerosols (particles) in the atmosphere and eventually form clouds.

The stratosphere is located above the troposphere and is a stable layer with few clouds and weather phenomena. The layer exhibits temperature inversions, with temperatures gradually increasing with altitude due to the absorption of ultraviolet light by the ozone layer. The top of the stratosphere is bounded by the stratopause, which is approximately 30 km above sea level. This boundary marks the transition to the next atmospheric layer, the mesosphere.

The mesosphere is located about 30-80 km above the Earth's surface. In this layer, the temperature decreases as the altitude increases. The upper part of the mesosphere has almost no water vapor and ionized particles, so it forms the ionosphere with the thermosphere and exosphere above.

In the ionosphere, solar radiation impacts the atmosphere with a power density of 1370W/m^2, and the spectrum of radiation frequencies extends from radio to X-ray frequencies (Böhm et al., 2013). Ultraviolet photons, or photons of shorter wavelength, are capable of ionizing atmospheric atoms or molecules by detaching electrons during the collision. Such ionization events occur when solar radiation collides with gas atoms or molecules, which then absorb a portion of the radiation, liberating free electrons and creating positively charged ions. While photoionization of electromagnetic radiation is the primary source of ionization, ionization can also occur through interactions with energetic particles in the solar wind and cosmic rays, albeit on a much smaller scale (Moses, 2004). The ionized state of the ionosphere affects the propagation of electromagnetic waves, a phenomenon known as ionospheric refraction. This refraction is a key factor affecting the performance of spaceborne microwave geodetic techniques.

2.2. Ionospheric delay in spaceborne microwave geodesy

When electromagnetic waves propagate in a medium, they will be refracted, which affects their propagation speed and propagation path. In geodesy, the term 'delay' commonly refers to the difference between the actual propagation distance and the geometric distance due to refraction effects. Specifically, according to Fermat's principle (Born et al., 2000), the propagation distance s^{ion} can be expressed as

$$s^{ion} = \int_{S^{ion}} n_{ion} ds, \tag{2.1}$$

where S^{ion} is the actual path of the signal through the ionosphere, and n_{ion} is the refractive index of the ionosphere. And the geometric distance s_0^{ion} can be expressed as

$$s_0^{ion} = \int_{S_0^{ion}} ds_0, \tag{2.2}$$

where S_0^{ion} is the straight path of the signal through the ionosphere. The ionospheric delay Δs^{ion} then can be defined as

$$\Delta s^{ion} = s^{ion} - s_0^{ion} = \int_{S^{ion}} n_{ion} ds - \int_{S_0^{ion}} ds_0. \tag{2.3}$$

The atmosphere in the ionosphere is in a plasma state, so the refractive index n_{ion} in which electromagnetic waves propagate can be described by the Appleton–Hartree equation (Helliwell, 1966), that is:

$$n_{ion}^2 = 1 - \frac{X(1-X)}{1 - X - \frac{1}{2}Y^2\sin^2\theta \pm \left(\left(\frac{1}{2}Y^2\sin^2\theta\right)^2 + (1-X)^3Y^2\cos^2\theta\right)^{1/2}} \quad (2.4)$$

where $X = \frac{\omega_i^2}{\omega^2}$, $Y = \frac{\omega_H}{\omega}$, $\omega_0 = 2\pi f_0 = \sqrt{\frac{N_e e^2}{\epsilon_0 m_e}}$, $\omega_H = 2\pi f_H = \frac{B_0|e|}{m_e}$; $\omega = 2\pi f$ denotes angular frequency, ω_0 denotes electron plasma frequency, ϵ_0 denotes permittivity of free space, θ denotes angle between the ambient magnetic field vector and the wave vector, N_e denotes electron density, f denotes wave frequency, ω_H denotes electron gyro frequency, B_0 denotes ambient magnetic field strength, e denotes electron charge and m_e denotes electron mass. According to Tucker & Fannin (1968) and Hartmann & Leitinger (1984), assuming that the magnetic field is associated with the propagation direction, and sin $\theta \approx 0$, the refractive index n_{ion} can be approximated as:

$$n_{ion} = 1 - \frac{X}{2} \pm \frac{XY}{2}\cos\theta - \frac{X^2}{8}. \quad (2.5)$$

Following Brunner & Gu (1991), the term X and Y can be approximated as constants C_X and C_Y as

$$C_X \equiv \frac{e^2}{4\pi^2\varepsilon_0 m_e} = 80.62, \quad (2.6)$$

$$C_Y \equiv \frac{\mu_0 e}{2\pi m_e}, \quad (2.7)$$

where μ_0 is the permeability in vacuum. So the Eq. 2.5 can be expressed in order of f^{-1} as:

$$n_{ion} = 1 - \frac{C_X}{2}N_e f^{-2} \pm \frac{C_X C_Y}{2}N_e B_0\cos\theta f^{-3} - \frac{C_X^2}{8}N_e^2 f^{-4}. \quad (2.8)$$

By substituting Eq. 2.8 into Eq. 2.3, the ionospheric delay Δs_{ion} then become

$$\Delta s^{ion} = -\frac{C_X}{2f^2}\int_{S^{ion}} N_e\, ds \pm \frac{C_X C_Y}{2f^3}\int_{S^{ion}} N_e\, B_0\cos\theta\ ds - \frac{C_X^2}{8f^4}\int_{S^{ion}} N_e^2\, ds + \kappa, \quad (2.9)$$

where $\kappa = \int_{S^{ion}} ds - \int_{S_0^{ion}} ds_0$ is the curvature effect. For simplicity, when the curvature effect is ignored, $ds = ds_0$ and $\kappa = 0$, in this situation, the ionospheric delay Δs^{ion} then become

$$\Delta s^{ion} = -\frac{C_X}{2f^2}\int_{S_0^{ion}} N_e\, ds_0 \pm \frac{C_X C_Y}{2f^3}\int_{S_0^{ion}} N_e\, B_0\cos\theta\ ds_0 - \frac{C_X^2}{8f^4}\int_{S_0^{ion}} N_e^2\, ds_0. \quad (2.10)$$

According to Eq. 2.10, the ionospheric delay can be divided into first-order term (f^{-2}), second-order term (f^{-3}) and third-order term (f^{-4}), and the first-order term usually accounts for more than 99% of the total ionospheric delay. E.g. for the GPS L1 band (f = 1575.42 MHz), the first-order delay is at ten meters level, and the second-order and third-order delays are at centimeter and micrometer level (Hoque & Jakowski, 2010). Even during the peak period of solar activity, the second-order ionospheric delay at low altitudes usually does not exceed 12 cm, and the third-order ionospheric effect usually does not exceed 6 mm (Teunissen & Montenbruck, 2017). Therefore, in order to simplify the calculation, only the first-order ionospheric delay effect is usually considered in general applications. For the first-order ionospheric delay

$$\Delta s_{(1)}^{ion} = -\frac{C_X}{2f^2}\int_{S_0^{ion}} N_e\, ds_0, \quad (2.11)$$

define the constant factor as:

$$C_2 = \frac{C_X}{2} = 40.31, \tag{2.12}$$

so the first-order delay can be seen as the liner function of Slant Total Electron Content (STEC) and signal frequency. The STEC is the total number of free electrons integrated along a tube of one-meter squared cross-section in signal propagation path, unit in 10^{16} electrons/m^2:

$$STEC = \int_{S_0^{ion}} N_e \, ds_0. \tag{2.13}$$

There are two methods to eliminate ionospheric delay. The first is to use the TEC, expressed as the number of free electrons in a column with a cross-sectional area of one meter square, to estimate the vertical delay, and then convert it into slant direction using a mapping function. Since the 1980s, various TEC models have been proposed to provide TEC estimation products ($\mathrm{TEC_{model}}$), such as: Klobuchar Model (Klobuchar, 1986), NeQuick Model (Giovanni & Radicella, 1990), the International Reference Ionosphere (IRI) Model (Bilitza et al., 2011), and the Global Assimilative Ionospheric Model (GAIM) (Schunk et al., 2004). By using the Single-Layer Model for the ionosphere (SLM), as shown in Fig. 2.1, and its mapping function, the $\mathrm{TEC_{model}}$ can be mapped to STEC estimation to calculate the ionospheric delay. In SLM it is assumed that all free electrons are concentrated in an infinitesimally thin layer above the Earth's surface (Schaer, 1999). The altitude H of this layer is usually set between 350 and 500 km, slightly above the altitude where the highest electron density is expected. The signal transmitted from the satellite to the receiver crosses the ionospheric shell in the so-called ionospheric pierce point (IPP). The zenith angle at the IPP z' and the signal arrives at the ground station with zenith angle z as shown in Fig. 2.1.

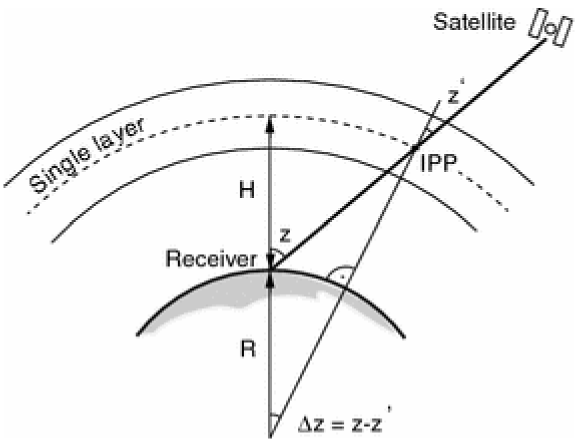


Fig. 2.1 Single-layer model for the ionosphere (Todorova et al.., 2008)

From the geometric relationship, it can be derived:

$$\sin z' = \frac{R}{R+H} \sin z, \tag{2.14}$$

where R is the mean Earth radius and H is the height of the single layer in km. Therefore, the mapping function $F(z)$ which convert TEC to STEC can be defined as:

$$F(z) = \frac{1}{\cos(z')} = \frac{1}{\sqrt{1-\sin^2 z'}}. \tag{2.15}$$

And the first-order ionospheric delay estimated by TEC model then can be written as:

$$\Delta s^{ion} \approx \Delta s_{(1)}^{ion} = -\frac{40.31}{f^2}\frac{1}{\sqrt{1-\sin^2 z'}}\mathrm{TEC}_{\mathrm{model}}. \tag{2.16}$$

While the models mentioned above can provide an estimation of TEC, the accuracy of these models is influenced by a multitude of factors. These include sunspot activity, seasonal and diurnal variations, the paths of signal propagation, and alterations in the location of observation points. Consequently, procuring precise estimates of TEC poses a significant challenge. Eq. 2.10 demonstrates that the ionospheric delay is correlated with the signal frequency. This correlation offers the potential for geodetic techniques supporting multi-frequency observations, such as GNSS, to eliminate the ionospheric delay by employing the signal combination at different frequencies. Since ionosphere-free GNSS signals are a prerequisite for accurately solving the tropospheric delay in the subsequent chapters, the following provides a brief overview of GNSS Earth observation technology and the strategy for eliminating ionospheric effects through the combination of dual-frequency GNSS signals. Currently, there are four GNSSs in operation, which are: GPS (US), GLONASS (Russia), BeiDou (China), and Galileo (EU). Irrespective of the GNSS variant, the satellites use signals across multiple frequencies. The details of these frequencies are presented in Table 2.1.

Table 2.1 The frequency of in-operation GNSSs

GPS	**GLONASS**	**BeiDou**	**Galileo**
L1 1575.42 MHz	L1 1602–1615.5 MHz	B1 1561.098 MHz	E1 1575.46 MHz
L2 1227.60 MHz	L2 1246–1256.5 MHz	B2 1207.14 MHz	E6 1278.75 MHz
L5 1176.45 MHz	L3 1202.025 MHz	B3 1268.52 MHz	E5 1191.795 MHz

By using dual-frequency GNSS signals, the dual-frequency GNSS observation equation can be established as follows:

$$\begin{aligned}
P_{L_1} &= \mathrm{s} + c(\delta t_R - \delta t^S) + \Delta \mathrm{s}^{trop} + \Delta s^{ion}(L1) + c(b_R - b^S)_{L1} + \mathrm{r},\\
P_{L_2} &= \mathrm{s} + c(\delta t_R - \delta t^S) + \Delta \mathrm{s}^{trop} + \Delta s^{ion}(L2) + c(b_R - b^S)_{L2} + \mathrm{r},\\
\varphi_{L_1} &= \mathrm{s} + c(\delta t_R - \delta t^S) + \Delta \mathrm{s}^{trop} - \Delta s^{ion}(L1) + \lambda_{L_1} B_{L_1} + \mathrm{r},\\
\varphi_{L_2} &= \mathrm{s} + c(\delta t_R - \delta t^S) + \Delta \mathrm{s}^{trop} - \Delta s^{ion}(L2) + \lambda_{L_2} B_{L_2} + \mathrm{r},
\end{aligned} \tag{2.17}$$

where P_{L_1}, P_{L_2} are the received pseudorange code in L1 and L2 frequency; $\varphi_{L_1}, \varphi_{L_2}$ are the received signals of carrier phase in L1 and L2 frequency; s is geometric distance between receiver and satellite; $\delta t_R, \delta t^S$ are receiver and satellite clock offsets to the GNSS system time; Δs^{trop} is the delay caused by troposphere, the detail of this term will be introduced in next section; $\Delta s^{ion}(L1), \Delta s^{ion}(L2)$ are ionospheric delay with frequency L1 and L2; b_R, b^S are frequency-dependent hardware delays of the satellite and receiver (in ns); $\lambda_{L_1}, \lambda_{L_2}$ are the wavelength of L1 and L2 bands, B_{L_1}, B_{L_2} are the integer carrier phase ambiguity and hardware biases which need to be estimated; r is the random error. Then a liner combination can be established as:

$$\begin{aligned}
P_{1,2} &= n_1 P_{L1} + n_2 P_{L2},\\
\varphi_{1,2} &= n_1 \varphi_{L1} + n_2 \varphi_{L2},
\end{aligned} \tag{2.18}$$

The objective is to eliminate ionospheric delay, therefore, the combination should fulfill:

$$n_1 \Delta s^{ion}(L1) + n_2 \Delta s^{ion}(L2) = 0. \tag{2.19}$$

When the n_1, n_2 are:

$$n_1 = +\frac{f_{L_1}^2}{f_{L_1}^2 - f_{L_2}^2}, \qquad n_2 = -\frac{f_{L_2}^2}{f_{L_1}^2 - f_{L_2}^2}, \tag{2.20}$$

and the ionospheric delay has been approximated by first-order, as Eq. 2.16 showed, Eq. 2.19 has been fulfilled. From this, it can be obtained the "ionospheric-free" observation equation without using TEC estimation:

$$\begin{aligned} P_{1,2} &= n_1 P_{L1} + n_2 P_{L2} \\ &= \mathrm{s} + c(\delta t_R - \delta t^S) + \Delta \mathrm{s}^{trop} + n_1 c(b_R - b^S)_{L1} + n_2 c(b_R - b^S)_{L2} + \mathrm{r}, \end{aligned}$$

$$\varphi_{1,2} = n_1 \varphi_{L1} + n_2 \varphi_{L2} = \mathrm{s} + c(\delta t_R - \delta t^S) + \Delta \mathrm{s}^{trop} + n_1 \lambda_{L_1} B_{L_1} + n_2 \lambda_{L_2} B_{L_2} + \mathrm{r}. \tag{2.21}$$

It is important to note that the description "ionosphere-free" is not entirely correct as it does not take into account the effects of higher-order ionospheric errors or curvature. Brunner & Gu (1991) indicated that the higher-order terms as well as the curvature effects, are less than 0.1% of the total value in L-band. For an in-depth exploration of high-order ionospheric correction methodologies, please refer to the following papers: Brunner & Gu (1991), Hoque & Jakowski (2008).

2.3. Tropospheric delay in spaceborne microwave geodesy

After passing through the ionosphere, microwave signals need to pass through the neutral atmosphere (also called the neutrosphere) to reach the surface of the earth. In the neutrosphere, the propagation of the signal will be delayed just like in the ionosphere. However, the delay in the neutrosphere does not depend on the frequency, so it is not possible to use a dual-frequency combination to eliminate the tropospheric delay like in the ionosphere. Therefore, the strategy to eliminate the neutrosphere delay usually relies on meteorological physical models. Before introducing those models, it is essential to clarify the term "tropospheric delay". In fact, within the field of geodesy, "tropospheric delay" covers the cumulative signal delays within the neutral atmosphere, which includes the troposphere, stratosphere, and mesosphere. Since the tropospheric delay constitutes the main component, the neutrosphere delay is commonly referred to as "tropospheric delay". Unless otherwise specified and in accordance with geodetic conventions, the term "troposphere" used in subsequent sections of this article will represent the neutral atmosphere.

Since electromagnetic waves will be refracted when passing through the troposphere, and the propagation speed of the signal in the atmosphere is lower than the propagation speed of light in a vacuum, according to Fermat's principle, the tropospheric delay Δs^{trop} can be expressed in a form similar to Eq. 2.3:

$$\Delta s^{trop} = s^{trop} - s_0^{trop} = \int_{S^{trop}} n_{trop} ds - \int_{S_0^{trop}} ds_0, \tag{2.22}$$

where S^{trop}, S_0^{trop} are the actual propagation path and straight path of signal through troposphere, and n_{trop} is the refractive index of troposphere. In the troposphere, the refractive index n_{trop} is very close to 1, thus, it is convenient to use the refractivity *N* (in "N-units", mm/km, or ppm) instead n_{trop} as:

$$N = 10^6 (n_{trop} - 1). \tag{2.23}$$

In the troposphere, the abundance of hydrostatic atmospheric components is very uniform and constant. The wet part, water vapor, is the only component with a significant dipole moment that can affect the propagation of electromagnetic waves. Therefore, based on the physical properties of the tropospheric components, ignoring the contribution of liquid water only in the millimeter range, the tropospheric refractivity can be expressed as a function of pressure, temperature, and humidity as:

$$N = k_1 \frac{p_\mathrm{d}}{T} Z_\mathrm{d}^{-1} + k_2 \frac{p_\mathrm{w}}{T} Z_\mathrm{w}^{-1} + k_3 \frac{p_\mathrm{w}}{T^2} Z_\mathrm{w}^{-1} \equiv K_1 \frac{p_\mathrm{d}}{T} + K_2 \frac{p_\mathrm{w}}{T} + K_3 \frac{p_\mathrm{w}}{T^2}, \tag{2.24}$$

where p_d and p_w are the pressure of dry atmosphere and wet atmosphere, Z_d and Z_W are the compressibility factors for dry and wet atmosphere, T is the temperature, k_1, k_2, k_3 are the constants based on measurement data (Essen & Froome, 1951), and the capital K_1, K_2, K_3 is used for the formulation that does not include compressibility. In order to determine the constant, several laboratories have conducted measurements, such as: Boudouris (1963), Thayer (1974), and Bevis (1994). Rüeger (2002) summarized those measurement and provide the "best average" values for $K_1 = 77.695\, K/hPa$, $K_2 = 71.97\, K/hPa$, and $K_3 = 375406\, K^2/hPa$. Unfortunately, due to the unstable behavior of water vapor, the determination of the constants K_2 and K_3 is subject to uncertainty, especially for K3, which results in Root-Mean-Square fluctuations of the delay of more than 6 mm worldwide. The first term in Eq. 2.24 represents the refractivity caused by dry atmosphere while the sum of other terms in Eq. 2.24 represents the refractivity due to the wet portion (water vapor). From this it can separate the delay caused by the dry atmosphere, also called hydrostatic delay Δs_h^{trop}, and the delay caused by the wet air (water vapor), also called wet delay Δs_w^{trop}, as shown in Eq. 2.25 and Eq. 2.26.

$$\Delta s_h^{trop} = 10^{-6} \int_{S^{trop}} K_1 \frac{p_\mathrm{d}}{T} ds, \tag{2.25}$$

$$\Delta s_w^{trop} = 10^{-6} \int_{S^{trop}} K_2 \frac{p_\mathrm{w}}{T} + K_3 \frac{p_\mathrm{w}}{T^2}. \tag{2.26}$$

Therefore, Eq. 2.22 can be written as:

$$\begin{aligned} \Delta s^{trop} &= \int_{S^{trop}} n_{trop} ds - \int_{S_0^{trop}} ds_0 = 10^{-6} \int_{S^{trop}} N ds + \int_{S^{trop}} ds - \int_{S_0^{trop}} ds_0 \\ &= 10^{-6} \int_{S^{trop}} K_1 \frac{p_\mathrm{d}}{T} ds + 10^{-6} \int_{S^{trop}} K_2 \frac{p_\mathrm{w}}{T} + K_3 \frac{p_\mathrm{w}}{T^2} + \int_{S^{trop}} ds - \int_{S_0^{trop}} ds_0 \\ &= \Delta s_h^{trop} + \Delta s_w^{trop} + S - G, \end{aligned} \tag{2.27}$$

where S is the geometric length of the actual propagation path of the signal and G is the straight propagation length of the signal, term $S - G$ is the effect of bending. In space geodesy, observations are made at arbitrary azimuth and elevation angles, so it is necessary to know the hydrostatic and wet delay contributions in each observation direction in order to be able to remove the effects of tropospheric delays that bias the observations. In order to estimate the delay in each observation direction, a common approach is to model the tropospheric delay in the zenith direction and use a mapping function to convert it into slant direction. Therefore, as an important concept, the Zenith Total Delay (ZTD), the Zenith Hydrostatic Delay (ZHD), and the Zenith Wet Delay (ZWD) can be defined as:

$$ZTD = ZHD + ZWD, \tag{2.28}$$

$$ZHD = 10^{-6} \int_{h_s}^{h_{top}} K_1 \frac{p_\mathrm{d}(z)}{T(z)} dz, \tag{2.29}$$

$$ZWD = 10^{-6} \int_{h_s}^{h_{top}} K_2 \frac{p_\mathrm{w}(z)}{T(z)} + K_3 \frac{p_\mathrm{w}(z)}{T^2(z)} dz, \tag{2.30}$$

where h_s is the surface altitude of the observation point on the earth and h_{top} is the height of tropopause.

From Eq. 2.28 to Eq. 2.30, it can be known that the ZHD and ZWD are determined by the pressure and temperature at different heights. However, measuring temperature and pressure at any point in space is very challenging. To address this problem, NWM, such as products from the European Center for Medium-Range Weather Forecasts (ECMWF), have been proposed to provide gridded meteorological data estimates with a resolution of tens of kilometers. One of the state-of-the-art NWM, the fifth generation of European Reanalysis (ERA5) products (Hersbach et al., 2023), provides hourly stratified meteorological products with 37 pressure levels at a spatial resolution of 0.25°[1]. Using the meteorological data provided by NWM, it is possible to obtain centimeter-level ZTD estimates (Ding et al., 2023; Liu et al., 2022) for any location within the NWM coverage area. Assuming uniformity within each layer of the atmosphere as represented in ERA5, according to the trapezoidal integral method, the Eq. 2.29 and Eq. 2.30 can be written as:

$$\begin{aligned} ZHD_{ERA5} &= 10^{-6}\int_{h_s}^{h_{top}} K_1 \frac{p_{\mathrm{d}}(z)}{T(z)} dz \\ &= 10^{-6}\sum_{i=l_0}^{36}\frac{1}{2}K_1\left(\frac{p_i - p_{\mathrm{w}_i}}{T_i} + \frac{p_{i+1} - p_{\mathrm{w}_{i+1}}}{T_{i+1}}\right) d_{i,i+1}, \end{aligned} \tag{2.31}$$

$$\begin{aligned} ZWD_{ERA5} &= 10^{-6}\int_{h_s}^{h_{top}} K_2 \frac{p_{\mathrm{w}}(z)}{T(z)} + K_3 \frac{p_{\mathrm{w}}(z)}{T^2(z)} dz \\ &= 10^{-6}\sum_{i=l_0}^{36}\frac{1}{2}\left[K_2\left(\frac{p_{\mathrm{w}_i}}{T_i} + \frac{p_{\mathrm{w}_{i+1}}}{T_{i+1}}\right) + K_3\left(\frac{p_{\mathrm{w}_i}}{T_i^2} + \frac{p_{\mathrm{w}_{i+1}}}{T_{i+1}^2}\right)\right] d_{i,i+1}, \end{aligned} \tag{2.32}$$

where l_0 is the initial layer which corresponds to h_s. p_i and T_i are the pressure and temperature at i-th layer pressure level obtained from ERA5 directly. p_{w_i} is the water vapor pressure at i-th layer pressure level, which can be obtained by using Magnus–Tetens approximation (Alduchov & Eskridge, 1996):

$$p_{\mathrm{w}_i} = 6.1094\frac{RH_i}{100}\exp\left(17.625\frac{T_i - 273.15}{T_i - 273.15 + 243.04}\right), \tag{2.33}$$

where RH_i is the relative humidity at i-th layer pressure level. And $d_{i,i+1}$ is the geometric height difference between the altitude of i-th and i+1-th pressure level, h_{i+1} and the altitude of i-th pressure level h_i, which can be obtained from geopotential as follows:

$$d_{i,i+1} = h_{i+1} - h_i, \tag{2.34}$$

$$h_i = \frac{R * \frac{G_i}{g}}{R - \frac{G_i}{g}}, \tag{2.35}$$

where R is the radius of the Earth, G_i is the geopotential at i-th layer pressure level, and $g = 9.80665\,\mathrm{m/s^2}$ is the Earth's gravitational acceleration. If h_s does not correspond to any geometric height aligned with the 37-layer pressure levels in ERA5, a vertical interpolation given (Jade & Vijayan, 2008) can be used to interpolate T, e and p at l_0 layer by using the two nearest layers:

$$T_{l_0} = T_{l_0-1} + \frac{T_{l_0+1} - T_{l_0-1}}{h_{l_0+1} - h_{l_0-1}}\left(h_{l_0+1} - h_s\right), \tag{2.36}$$

$$p_{l_0} = \frac{w_{l_0-1}}{w_{l_0-1} + w_{l_0+1}} p_{h_s,l_0-1} + \frac{w_{l_0+1}}{w_{l_0-1} + w_{l_0+1}} p_{h_s,l_0+1}, \tag{2.37}$$

$$w_{l_0-1} = \frac{1}{\left(h_s - h_{l_0-1}\right)^2}; w_{l_0+1} = \frac{1}{\left(h_{l_0+1} - h_s\right)^2}, \tag{2.38}$$

1 Data can be accessed at https://cds.climate.copernicus.eu/cdsapp#!/dataset/reanalysis-era5-pressure-levels?tab=overview

$$p_{h_s,l_0-1} = p_{l_0-1}\left[1+\frac{8.419*10^{-5}(h_{l_0-1}-h_s)}{p_{l_0-1}{}^{0.1902884}}\right]^{-5.255303}, \tag{2.39}$$

$$p_{h_s,l_0+1} = p_{l_0+1}\left[1+\frac{8.419*10^{-5}(h_{l_0+1}-h_s)}{p_{l_0+1}{}^{0.1902884}}\right]^{-5.255303}. \tag{2.40}$$

It is noteworthy that the interpolation method for p_w follows Eq. 2.37 to Eq. 2.40 in the same manner. By using Eq. 2.31 to Eq. 2.40, the ZHD and ZWD at each grid point can be calculated. Subsequently, employ the distances between the target position and these four grid points to execute Inverse Distance Weighting interpolation (IDW) as follows:

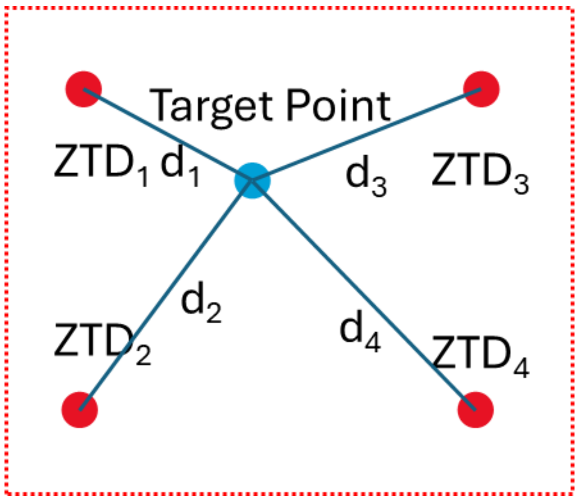


Fig. 2.2 Inverse Distance Weighting interpolation

$$ZTD_{target} = \sum_{i=1}^{4} w_i ZTD_i \tag{2.41}$$

$$w_i = \frac{d_i^{-2}}{\sum_{j=1}^{4} d_j^{-2}} \tag{2.42}$$

where ZTD_i denotes the ZTD estimation at grid point, and d_i is the distance between the target position and the corresponding grid point.

In addition to NMW, some empirical models, such as the Global Pressure and Temperature model 3/ Vienna Mapping Functions 3 (GPT3/VMF3) (Landskron & Böhm, 2018), can also provide estimates of temperature and pressure. However, these empirical models often only provide surface information. According to the ideal gas law, the compressibility factor of dry air Z_d can be written as

$$Z_d = 8.3145\frac{p_d M_d}{\rho T}, \tag{2.43}$$

where M_d is the molar mass of dry air, and ρ is the density of air. Therefore, the ZHD can be expressed as:

$$ZHD = 8.3145 * 10^{-6}\int_{h_s}^{h_{top}} k_1 \frac{p_d(z)}{T(z)}\frac{\rho(z)T(z)}{p_d(z)M_d}dz = 10^{-6}k_1\frac{8.3145}{M_d}\int_{h_s}^{h_{top}}\rho(z)dz. \tag{2.44}$$

From Eq. 2.44, it can be known that the ZHD only relevant with the density of air. From hydrostatic equilibrium

$$\frac{dp}{dz} = -\rho(z)g(z), \tag{2.45}$$

where $g(z)$ is the gravity along the vertical coordinate z, it can yield

$$p_s = \int_{h_s}^{h_{top}} \rho(z)g(z)dz. \tag{2.46}$$

Let

$$g_m = \frac{\int_{h_s}^{h_{top}} \rho(z)g(z)dz}{\int_{h_s}^{h_{top}} \rho(z)dz}, \tag{2.47}$$

the ZHD become

$$ZHD = 10^{-6}k_1 \frac{8.3145}{M_d} g_m^{-1} p_s. \tag{2.48}$$

By expanding $g(z)$ to the first order of z, it can be seen that g_m is very nearly to the acceleration due to the gravity at the center of mass of the vertical column. According to (Saastamoinen, 1972), g_m can be approximated as:

$$g_m = 9.8062(1 - 0.00266 * \cos(2LAT) - 0.28 * h_s * 10^{-6}), \tag{2.49}$$

where LAT is geodetic latitude. From this we obtain the famous Saastamoinen hydrostatic model:

$$ZHD = \frac{0.0022768 * p_{h_s}}{1 - 0.00266 * \cos(2LAT) - 0.28 * h_s * 10^{-6}}. \tag{2.50}$$

The Saastamoinen hydrostatic model is a fairly accurate model, with errors mainly coming from the measurement error of the surface pressure. Under typical meteorological conditions, the ZHD estimated by Saastamoinen model at sea level is about 2.3 m. A surface pressure error of 1 hPa results in an error of about 2.3 mm. When surface pressure measurements are not available, empirical models such as GPT3/VMF3 can provide surface temperature and pressure estimates at a spatial resolution of 1° every six hours. Since the altitude of the target position is often different from that of the grid points, when using the empirical model, it is necessary to adjust the four grid points around the target position to the same pressure level using the following formula:

$$p_{h_s} = p_g * \left(1 - 0.000026 * (h_s - h_g)\right)^{5.225}, \tag{2.51}$$

where p_g, h_g are the surface pressure and altitude at grid point. Then calculate the four grid points at the same pressure level using Eq. 2.50, and the ZHD at the target position can be obtained by IDW interpolation using Eq. 2.41 and Eq. 2.42.

The behavior of water vapor above the surface is often unpredictable due to the effects of turbulence. Therefore, it is extremely challenging to infer ZWD through empirical models of surface temperature and pressure. In the absence of real observational data, the estimation of ZWD usually follows NWM ray tracing, that is, Eq. 2.32 and its similar forms. Despite the use of state-of-the-art NWM, the ZWD retrieved by this method still has centimeter-level errors, so it is mainly used as a priori estimation or when there is no real observation data. It should be noted that some empirical models, such as VMF3, also provide estimates of surface ZWD in grid form. Since the grid points are at different altitudes from the target positions, they still need to be vertically adjusted before horizontal interpolation. Since the water vapor pressure approximately follows an exponential decay approximation, the vertical adjustment of ZWD follows the following method:

$$ZWD = ZWD_g * \exp\left(-\frac{h_s - h_g}{2000}\right), \tag{2.52}$$

where ZWD_g denotes the ZWD estimation at the grid point.

Once the estimates of ZHD and ZWD are obtained, assuming that the neutral atmosphere has azimuthal symmetry (i.e., at a constant elevation angle, the delay does not depend on the azimuth of the observation), the tropospheric delay Δs^{trop} can be obtained by combining ZHD and ZWD with their corresponding mapping functions, as described follows:

$$\Delta s^{trop} = ZHD * mf_h(\varepsilon) + ZWD * mf_w(\varepsilon), \tag{2.53}$$

where $mf_h(\varepsilon)$ and $mf_w(\varepsilon)$ are the mapping function for hydrostatic delay and wet delay at elevation angle ε. Since the height of the wet part of the troposphere is about 2 km and the height of the hydrostatic part is about 8 km (see Fig. 2.3), for most of the space geodetic scenes (elevation angle greater than 20°), the hydrostatic mapping function mf_h is less than the wet mapping function mf_w.

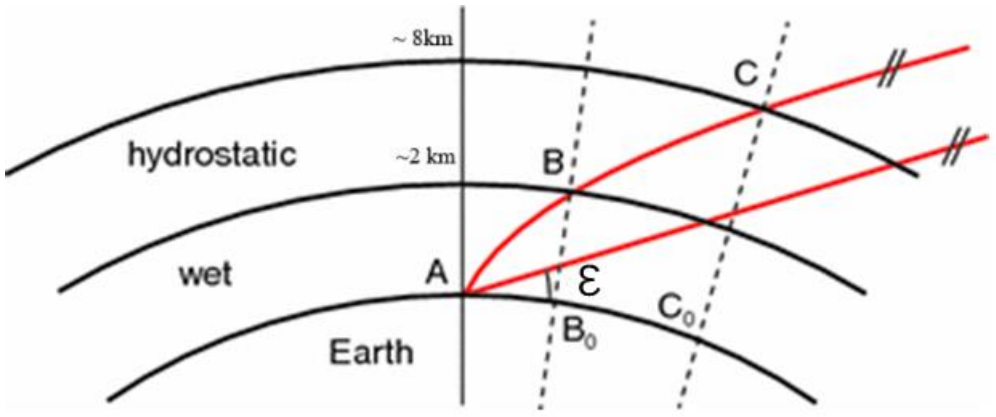


Fig. 2.3 The height of the wet part and hydrostatic part of troposphere (Nilsson et al., 2013). The mapping function describes the ratio of the slant path to the vertical distance. The hydrostatic mapping function $mf_h = \frac{AC}{CC_0}$ is less than the wet mapping function $mf_w = \frac{AB}{BB_0}$.

Marini (1972) first proposed the atmospheric refraction model based on this concept. With the improvement of Herring (1992), the following mapping function form has been widely accepted:

$$mf(\varepsilon) = \frac{1 + \frac{a}{1 + \frac{b}{1 + c}}}{\sin(\varepsilon) + \frac{a}{\sin(\varepsilon) + \frac{b}{\sin(\varepsilon) + c}}}, \tag{2.54}$$

where a, b, c are the coefficients that depend on integrals of refractivity through the atmosphere, which depends on the surface meteorological data, the location and the altitude. Although in the absence of NWM data, the factors a, b, c can be estimated by continuous empirical models using spherical harmonics functions of the Day of Year, such as the GMF function corresponding to GPT3, its ZHD may be coupled with the compensation of atmosphere, making its performance worse than the discrete models using ECMWF reanalysis data, such as VMF3 (Steigenberger et al., 2009). Therefore, in most cases, it is recommended to use the discrete model VMF3 to map the zenith delay into the slant direction.

The above methods describe the estimation of tropospheric delay when there is no real observation data. However, even with the best NWM (such as ERA5), the resolution of tens of kilometers and centimeter-level accuracy are still challenged to meet the requirements of high-precision geodetic tasks, such as surface deformation monitoring. Fortunately, the GNSS Precise Point Positioning (GNSS-PPP) technology provides the possibility to directly retrieve tropospheric delay. When there is a dual-frequency GNSS station at the target position, its ZTD can be directly solved by ionospheric-free GNSS observations using Eq. 2.21 combined with a mapping function. Leveraging the high-precision GNSS satellite orbit and clock products supplied by the International GNSS Service (IGS), GNSS-PPP mode can deliver station-wised ZTD with high accuracy and temporal resolution. Since the observations of GNSS stations come from continuous observations of multiple satellites, the ZWD solved by Eq. 2.21 using the least squares method is usually regarded as the reference or ground truth. In order to improve the accuracy and convergence speed of the solution, the ZHD provided by VMF3 (essentially the Saastamoinen hydrostatic model) and its corresponding dry mapping function $mf_h(\varepsilon)$ is usually accepted as the prior delay and deducted, and the actual solution is the remaining part, which is considered to represent the wet delay Δs_w^{trop}. Then the wet mapping function can be used to map the solved wet delay Δs_w^{trop} to the zenith direction to obtain the ZWD. The solving program of tropospheric delay retrieved by GNSS is usually implemented by GNSS-PPP solution software, such as Bernese GNSS software or GipsyX, by inputting the RENIX file generated by the receiver. In this thesis, the hardware of the GNSS receiver and the configuration of the software are not discussed in detail. For more information, please refer to (Teunissen & Montenbruck, 2017) and (Dach et al., 2015).

In addition, considering that the ZWD can be related to the water vapor above the surface, according to (Bevis, 1994), ZWD and Integrated Water Vapor (IWV) (also known as Precipitable Water Vapor or Total Column Water Vapor, in units of $\mathrm{kg/m^2}$, representing the mass of water vapor within a 1 $\mathrm{m^2}$ atmospheric column.) can be converted to each other according to the following equation:

$$\mathrm{IWV} = \Pi * \mathrm{ZWD}\,, \tag{2.55}$$

$$\Pi = \frac{10^6}{R_v * \left[K_2' + \frac{K_3}{T_m}\right]}\,, \tag{2.56}$$

$$T_m = \frac{\int_{h_s}^{h_{top}} \frac{p_\mathrm{w}(z)}{T(z)} dz}{\int_{h_s}^{h_{top}} \frac{p_\mathrm{w}(z)}{T^2(z)} dz}\,, \tag{2.57}$$

Where $R_v = 461.522 J/(kg \times K)$ (Kestin et al., 1984), $K_2' = 22.1\, K/hpa$ is the adjusted refractivity constant and T_m is the weighted mean temperature calculated by the integration ratio of water vapor pressure p_w and temperature T from the height of the surface to the tropopause. This equation means that ZWD can be measured by other water vapor detection technologies, such as Radiosonde or Radiometer. Similarly, ZWD retrieved by GNSS can also be used to invert the changes in water vapor in the atmosphere. Since ZWD retrieved by GNSS boasts outstanding characteristics, including high accuracy, high temporal resolution, and all-weather capability, GNSS water vapor retrieval has become a vital data source for precipitation forecasting, environmental change, extreme weather, and other related research.

It should be additionally pointed out that in addition to modeling ZHD and ZWD using the refractivity of the dry and wet atmosphere, there are other modeling methods for ZTD. Yu et al. (2018) proposed using the Iterative Tropospheric Decomposition (ITD) model to decompose the ZTD into an altitude-related stratified delay (SD) and an elevation-related turbulent delay (TD), as Eq. 2.58 shows:

$$ZTD = TD(x, y) + SD(h_s) + r_u, \tag{2.58}$$

$$SD(h_s) = SD_0 e^{-\beta \frac{h_s - h_{min}}{h_{max} - h_{min}}} \tag{2.59}$$

where x, y are the position in the local topocentric coordinate system, SD_0 is the stratified component delay at sea level, β is the coefficient factor, h_{min} and h_{max} are the Minimum and maximum altitudes of reference stations within 100 km (Yu et al., 2017)-150 km (Yu et al., 2018), and r_u is the unmodeled residual. The ITD model assumes that within 150 km, the stratified delay can be expressed in the form of an exponential function using the two coefficients L_0 and β obtained by regression, and only the spatially correlated turbulent delay component can be interpolated by using IDW. To determine the coefficient factors L_0,β and the turbulent delay component $TD(x, y)$ for IDW interpolation, the ITD model follows an iterative decomposition approach searching the parameters using the ZTD retrieved by GNSS or the grid ZTD derived from NWM ray tracing within 150 km of the target position. When the coefficient factor converges, the turbulent delay component at the target position is obtained through IDW interpolation based on the reference turbulence component TD_{ref} as shown in Eq. 2.60.

$$TD(x, y) = \sum_{i=1}^{n} w_{ti} TD_{ref}(x_i, y_i), \tag{2.60}$$

$$w_{ti} = \frac{d_i^{-2}}{\sum_{j=1}^{n} d_i^{-2}}, \tag{2.61}$$

where d_i is the distance between the target position and the position of the i-th reference. The iteration follows the following steps:

1. Assume that the turbulence component delay is zero, use all reference ZTDs to estimate coefficient factors L_0,β, and obtain the residual r_u.

2. Use the distance weight w_{ti} and the residual r_u to solve the reference turbulence delay component TD_{ref} of each reference ZTD according to Eq. 2.62.

$$\begin{bmatrix} TD_{ref,1} \\ \vdots \\ TD_{ref,n} \end{bmatrix} = \begin{bmatrix} 0 & w_{12} & \cdots & w_{1n} \\ w_{21} & 0 & \cdots & w_{2n} \\ \vdots & \vdots & 0 & \vdots \\ w_{n1} & \cdots & w_{n,n-1} & 0 \end{bmatrix} = \begin{bmatrix} r_{u,1} \\ r_{u,2} \\ \vdots \\ r_{u,n} \end{bmatrix}. \tag{2.62}$$

3. Subtract the obtained TD_{ref} from each reference ZTD, and repeat steps 1 and 2 until the coefficients converge.

4. When the coefficients converge, the ZTD_{ITD} at the target position can be calculated according to Eq. 2.58. The stratified component *SD* is calculated using Eq. 2.59 based on the converged coefficient factors L_0,β, while the turbulent component *TD* and the residual r_u at target position are obtained using IDW interpolation, as shown in Eq. 2.60 and 2.61.

Compared with the traditional interpolation method (Eq. 2.41, directly interpolating the ZTD estimated by four grid points around the target position), the ITD model decomposes the horizontal and vertical correlation components of the ZTD and only interpolates the horizontal correlation component. Yu et al. (2018) proposed that this interpolation method can provide more competitive tropospheric delay products than traditional interpolation methods and integrated them into an online service called Generic Atmospheric Correction Online Service for InSAR (GACOS)[2] for use in InSAR atmospheric correction applications. However, this method currently lacks a corresponding mapping function to map it to the slant direction. At the same time, this decomposition method cannot accurately reflect the delay caused by the difference in thickness of the hydrostatic and wet atmosphere (see Fig. 2.3), which may pose a challenge to the accurate estimation of tropospheric delay Δs^{trop}.

[2] Data can be accessed at http://www.gacos.net/

3. The fundamentals of Synthetic Aperture Radar Interferometry

This chapter presents the foundational concepts of surface deformation monitoring using InSAR. It covers topics such as the SAR&InSAR principle, Multi-temporal SAR Interferometry, and the processing workflow of the Stanford Method for Persistent Scatterers (StaMPS). A particular focus is placed on the scatterers' selection mechanism, including PS and DS. Given the scope of this study and space constraints, details regarding SAR imaging, processing, and calibration methods will not be included. For further information on these topics, readers are referred to the handbook *InSAR Principles: Guidelines for SAR Interferometry Processing and Interpretation* (Ferretti et al., 2007) and the ESA Sentinel Application Platform (SNAP) documentations (https://step.esa.int/main/doc/online-help/).

In this chapter, the content regarding PSI and the StaMPS is mainly derived from Hooper et al. (2007), and the content regarding Distributed Scatterers is mainly derived from Even & Schulz (2018) and Ferretti et al. (2011).

3.1. SAR&InSAR principle

InSAR technology has been in use since the late 1970s, following the introduction of spaceborne SAR systems. The first civilian SAR, Seasat, was launched in 1978 (Moreira et al., 2013), marking the widespread use of SAR in remote sensing and geodetic application. InSAR gained significant attention in 1991 with the European Space Agency's (ESA) launch of the ERS-1 satellite, which provided a large amount of SAR data. Since then, InSAR has steadily increased in importance, with more than 50 satellites currently orbiting the Earth, continuously acquiring data for scientific, governmental, and commercial applications. Prominent examples of these systems include Sentinel-1, TerraSAR-X, TanDEM-X, Cosmo-SkyMed, RADARSAT-2, ALOS II, SAOCOM, PAZ, etc.

Data collected from these satellites serves various purposes, including the monitoring of land, ice, and oceans, as well as the mapping and detection of environmental changes. Applications range from land cover classification and ocean current mapping to providing intelligence and situational awareness during natural disasters, such as mapping floods or disaster-affected areas. One of the most outstanding features of spaceborne SAR is its ability to acquire interferometric data for geodetic purposes over large areas. Specialized missions like SRTM and TanDEM-X have generated global Digital Elevation Models (DEM), which are very useful for a variety of studies. These datasets enable glaciologists to examine the extent, flow, and mass balance of glaciers and ice sheets (Arigony-Neto et al., 2007), climatologists to estimate forest biomass (Merchant et al., 2022), and geologists to investigate tectonic processes, volcanic activity, and earthquakes (Moon et al., 1998).

SAR systems act as active sensors, using radar technology (radio detection and ranging) to transmit microwave pulses and receive backscattered signals from the Earth's surface. A radar instrument mounted on a moving platform (usually airborne or satellite-based, but there are also vehicle-based and ground-based SAR for different applications) produces a two-dimensional radar backscatter map. In real aperture imaging radar systems, the azimuth resolution, which corresponds to the direction of radar movement, is proportional to the angular beamwidth of the antenna. Higher resolution can be achieved by increasing the antenna size. In a SAR system, the sensor moves along its trajectory (azimuth direction) and emits coherent microwave pulses at high pulse repetition frequencies, allowing scatterers to be illuminated multiple times during data acquisition and using a short physical antenna to simulate a long virtual antenna. As the sensor moves, the variation in distance between the sensor and the scatterer results in a radial velocity between both, which induces a Doppler shift in the recorded data. Compression techniques use this shift to focus the data in the azimuth direction, allowing advanced spaceborne SAR systems, such as Sentinel-1 and TerraSAR-X, to produce radar images with spatial resolutions down to the meter or sub-meter scale (for staring spotlight mode, the resolution is 24 cm).

In SAR imagery, the basic observation z for each pixel from the Single Look Complex (SLC) image of SAR is derived from the sum of the backscattered complex signals of all surface scatterers within that ground resolution cell. This is expressed in terms of two components: the amplitude A and the phase shift ϕ between the emitted and received signals, commonly referred to as phase, as described in Eq. 3.1.

$$z = Ae^{i\phi}. \tag{3.1}$$

According to Hanssen (2001), the amplitude A represents the magnitude of the backscattered signal. The phase ϕ, on the other hand, corresponds to the distance D between the SAR sensor and the surface target and the radar wavelength λ, as described below:

$$\phi = -\frac{4\pi D}{\lambda} + \phi_{scat} + \phi_{atmo} + 2k\pi, \qquad k \in \mathbb{N}, \qquad \phi \in [-\pi, \pi) \tag{3.2}$$

where ϕ_{scat} is the phase change caused by the scattering characteristics of the target and is close to zero for metal objects and may largely differ from zero in case of absorption, and ϕ_{atmo} is the phase delay due to atmospheric effects in the troposphere and ionosphere, as discussed in Chapter 2.

In geodesy, the main interest is to detect the motion of the ground over time. If the Earth's surface moves between two SAR satellite acquisitions, the D of the two acquisitions will be different, resulting in a corresponding phase shift. By analyzing this phase shift in space and time, surface deformation can be effectively monitored. This concept forms the core principle of InSAR, as shown in Fig. 3.1.

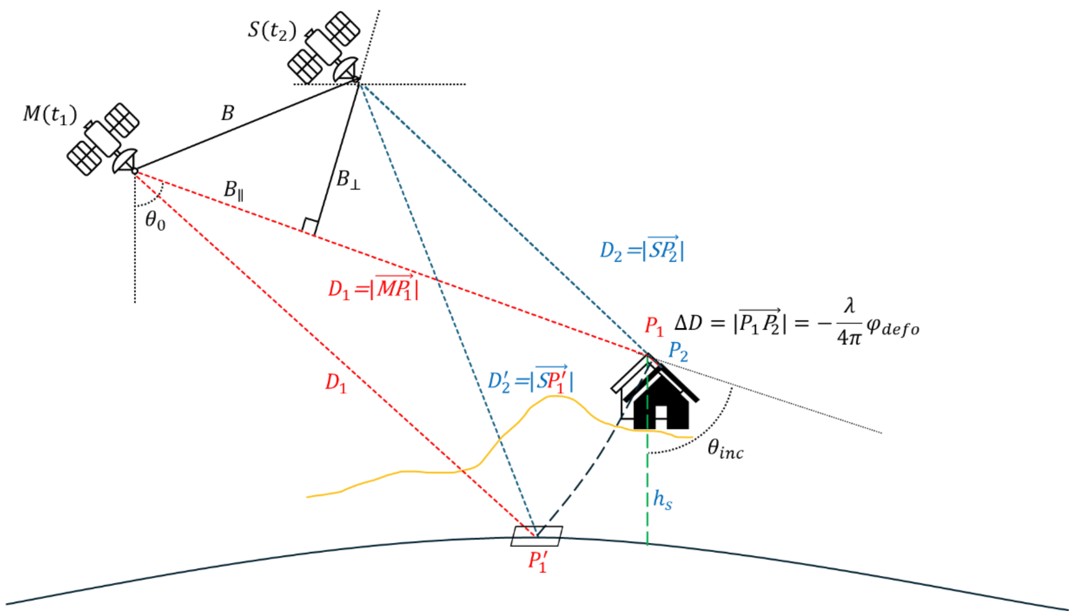

$S(t_2)$
$M(t_1)$
B
$B_{\perp}$
$B_{\parallel}$
θ_0
$D_2 = |\overrightarrow{SP_2}|$
$D_1 = |\overrightarrow{MP_1}|$
P_1 $\Delta D = |\overrightarrow{P_1P_2}| = -\frac{\lambda}{4\pi}\varphi_{defo}$
P_2
D_1
$D_2' = |\overrightarrow{SP_1'}|$
θ_{inc}
h_s
P_1'

Fig. 3.1 The geometric model of SAR interferometry. M and S are the satellite positions corresponding to the first (also called Master) acquisition at time t_1 and the second (also called Slave) acquisition at time t_2. P_1 is the position of the surface target at time t_1 and P_2 is the position of the surface target at time t_2. h_s is the height of the surface target above the reference surface. D_1 is the distance between M and P_1, and D_2 is the distance between S and P_2. P_1' is the position on the reference surface with the same range D_1 from M, corresponding to the pixel position in the SAR image. D_2' is the distance between S and P_1', its difference with D_1 can be used to calculate the flat earth phase. B is the baseline of those two acquisitions, $B_{\perp}$ and $B_{\parallel}$ are the perpendicular and the parallel component of the baseline with regard to the line-of-sight (LOS) direction of the Master. Due to the curvature of the earth and the local topography, the sensor's look angle θ_0 and the incidence angle θ_{inc} are not equal. ΔD is the displacement along LOS direction.

For two acquisitions z_1 and z_2, the interferogram is obtained by multiplying the complex conjugates of the two acquisitions

$$z_1 z_2^* = A_1 A_2 e^{i(\phi_1 - \phi_2)} = A_1 A_2 e^{i(\Delta\phi)}. \tag{3.3}$$

And the phase difference, also denoted as interferometric phase $\Delta\phi$, can be decomposed into the following components:

$$\Delta\phi = W\left(\varphi_{flat} + \varphi_h + \varphi_{defo} + \varphi_{atmo} + \varphi_{orbit} + \varphi_{noise}\right), \tag{3.4}$$

where φ_{flat} is the flat earth phase, caused by range differences of both orbits to a reference surface like the ellipsoid, as illustrated in Fig. 3.1 and can be expressed as:

$$\varphi_{flat} = -\frac{4\pi}{\lambda}(D_1 - D_2'), \tag{3.5}$$

Due to $B \ll D_1$, the range differences $D_1 - D_2'$ can be approximated as $B_{\parallel}$ by using far-field approximation (Zebker & Goldstein, 1985), therefore, the earth flat phase can be written as:

$$\varphi_{flat} = -\frac{4\pi}{\lambda} B_{\parallel}. \tag{3.6}$$

φ_h is the phase caused by topography h_s regarding to the reference surface, it can be defined as:

$$\varphi_h = -\frac{4\pi}{\lambda}\left(D_1 - D_2 - (D_1 - D_2')\right) = -\frac{4\pi}{\lambda}(D_2' - D_2). \tag{3.7}$$

Applying the far-field approximation again, $D_2' - D_2$ can be approximated as $\frac{B_\perp}{D_1 sin\theta_0} h_s$, and the topographic phase can be written as:

$$\varphi_h = -\frac{4\pi}{\lambda}\frac{B_\perp}{D_1 sin\theta_0} h_s. \tag{3.8}$$

$\varphi_{atmo}, \varphi_{orbit}, \varphi_{noise}$ are the phase contributions caused by atmospheric heterogeneities, satellite orbit error and noise including ϕ_{scat} difference between two acquisitions, which are spatially correlated and can be filtered out by using a spatial filter. The remaining part φ_{defo}, reflects the displacement:

$$\varphi_{defo} = -\frac{4\pi}{\lambda}\Delta D, \tag{3.9}$$

where ΔD is the displacement projected into the LOS direction. As the interferometric phase $\Delta\phi$ is wrapped into the interval $[-\pi, \pi]$, it cannot be directly decomposed into the combination of the above phases because of the unknown ambiguity k. Let W denote the wrapping operator, which maps the phase into $[-\pi, \pi)$:

$$W(\varphi) = \varphi - 2k\pi, \qquad k \in \mathbb{N}. \tag{3.10}$$

The process of estimating the ambiguity factor k is referred to as phase unwrapping. Unlike GNSS observation equations, InSAR cannot resolve phase ambiguities through multi-satellite observations, as it acquires data from only one satellite at a time. However, InSAR does generate large-scale, spatially mostly continuous interferograms, and its phase ambiguities are typically estimated using optimization algorithms that leverage this spatial continuity, like SNAPHU (Chen & Zebker, 2002).

The unwrapping process is ambiguous, which means that it is easy to wrap the model-generated phases like φ_{flat} and φ_h, or filter-generated phase like $\varphi_{atmo}, \varphi_{orbit}, \varphi_{noise}$ and subtract them from the interferometric phase $\Delta\phi$ to get the wrapped deformation phase ϕ_{defo} then do the unwrapping, but hard to unwrap $\Delta\phi$ first then to estimate $\varphi_{defo}, \varphi_{atmo}, \varphi_{orbit}$ and φ_{noise}. When the wrapped phase $\Delta\phi$ contains large atmospheric phase differences caused by weather events, complicated topographic phases, or deformation phases with large gradients, attempts to unwrap $\Delta\phi$ will result in severe unwrapping errors and lead to unwrapping failure. Therefore, an important step in monitoring surface deformation using InSAR is to accurately model and remove the non-deformed phase components. When long-term surface deformation monitoring is required, modeling and removing these non-deformation phases becomes a particularly challenging task. To address this issue, various MT-InSAR algorithms have been developed, which will be introduced in the next section.

3.2. Multi-Temporal InSAR

From Eq. 3.4, it can be seen that when $\varphi_{flat}, \varphi_h, \varphi_{atmo}$ and φ_{orbit} are all modeled well and eliminated, the accuracy of deformation monitoring depends on the level of the noise phase φ_{noise}. φ_{noise} mainly includes the change of scattering characteristics ϕ_{scat} between two acquisitions, processing error (such as coregistration error) and thermal noise of the sensor. In fact, for some land cover types, such as complex vegetation or water, the change of scattering characteristics between two acquisitions can change so much that the contribution of φ_{noise} dominates $\Delta\phi$. This phenomenon is called decorrelation. Decorrelation leads to a loss in the magnitude of the complex coherence γ between two SAR signals z_1 and z_2, which is defined as follows:

$$\gamma = \frac{E\{z_1 z_2^*\}}{\sqrt{E\{|z_1|^2\}E\{|z_2|^2\}}}, 0 \le \gamma \le 1. \tag{3.11}$$

where $E\{.\}$ denotes the expectation value. In practical situations, the assumption is made that it is possible to exchange the ensemble averages with spatial averages, obtained over a limited area surrounding the pixel of interest. This assumption is used to obtain the maximum likelihood estimator of the coherence magnitude $|\hat{\gamma}|$ for W surrounding pixels (Seymour & Cumming, 1994):

$$|\hat{\gamma}| = \frac{\sum_W z_1 z_2^*}{\sqrt{\sum_W |z_1|^2 \sum_W |z_2|^2}}. \tag{3.22}$$

In practice, not all pixels in the interferogram have sufficient correlation properties; only pixels that show sufficient coherence can ensure the validity of the unwrapped phase for displacement measurement. Therefore, identifying pixels with sufficient coherence is a fundamental step in InSAR processing. In addition, when analyzing deformation time series using MT-InSAR, it is particularly important to identify coherent pixels between different interferograms in the data stack.

For MT-InSAR, the analyzable pixels need to have good enough coherence in the data stacks formed by N interferograms, therefore the phase coherence $|\hat{\gamma}_\phi|$ (abbreviated as γ) then be introduced by Ferretti et al (2001) as:

$$\gamma = |\hat{\gamma}_\phi| = \frac{1}{N}\sum_{i=1}^{N} e^{i\Delta\phi_n}. \tag{3.23}$$

where $\Delta\phi_n$ is the wrapped residual phase of $\Delta\phi$ subtract modeled phase, . Combined with Eq. 3.4, the available scatterers can be obtained by modeling and removing components other than φ_{noise} to calculate coherence γ and select pixels with good coherence. The workflow of InSAR deformation analysis is shown below:

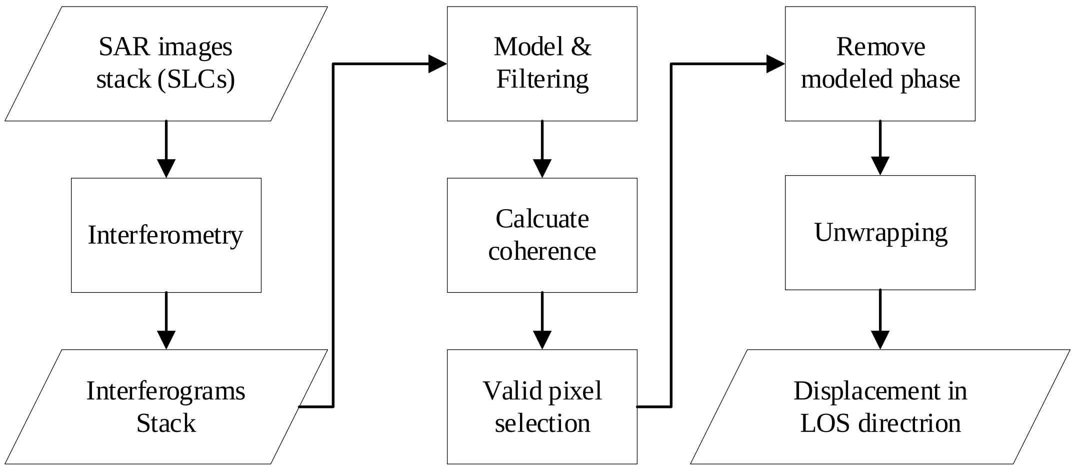


Fig. 3.2 The workflow of InSAR deformation analysis.

From the perspective of InSAR processing, scatterers that can be used for displacement analysis can be divided into two categories: PS and DS, as shown in Fig. 3.3. PS refers to stable, dominant scatterers within a resolution unit, such as trihedral artificial structures, buildings, or single rocks. The SAR echo signal, in this case, is primarily composed of the backscatter from this scatterer, with very small temporal decorrelation. In contrast, DS pixels lack dominant scatterers, resulting in the random behavior of multiple scatterers within the resolution unit. This randomness leads to larger phase variations at different acquisition times. Most DS exist on natural surfaces, such as forests, farmlands, exposed soil, and rock surfaces.

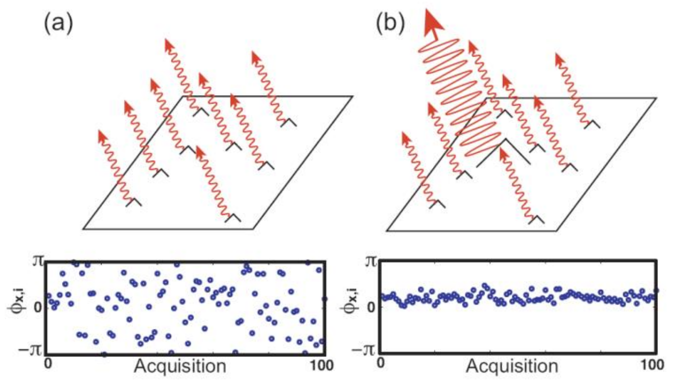


Fig. 3.3 Simulation of phase variation for (a) DS, and (b) PS (Hooper et al., 2007).

For PS, due to their stable phase, one SAR image can be selected from a set of N images as the master image, while the remaining N-1 images are used as slave images for the interferometric analysis. These slave images are registered with the master image to form N-1 interferograms. The selection of the master image is not a trivial task since any noise in the master image will affect the interferometric phase of all slave images. Therefore, the master image should be selected to minimize the total decorrelation of all interferograms (Hooper et al., 2007). Another important step is the subtraction of known phase contributions, in particular, the flat-earth phase φ_{flat} and the topographic phase φ_h. With the help of Eqs. 3.6 and 3.8, these phases can be simulated as accurately as possible using orbital information and DEM. However, the residual error $\Delta\varphi_{topo}$ due to DEM inaccuracies remain in each interferogram (also regards as look-angle error), and this error need to be estimated and subtracted by the subsequent processing algorithms as parameters to be determined. After removing the atmospheric phase φ_{atmo} and the estimated topographic phase residuals $\Delta\varphi_{topo}$, the phase for each PS point will be subtracted by a reference phase from a reference point or area, this will yield the displacement relative to the reference position. By performing the above operation on each interferogram in the stack, the relative displacement time series of the slave acquisitions relative to the master acquisition can be obtained. This technique is known as PSI. Since the PSI method focuses solely on PS with very high coherence, the only main parameters that need to be estimated are $\Delta\varphi_{topo}$ and φ_{atmo}. PSI has the advantages of few estimated parameters, high stability, and high computational efficiency, making it very suitable for accurately detecting the displacement of point targets. As a result, it is widely applied in geophysical and geodetic research. In the following section, the detailed processing flow of the Stanford Method for Persistent Scatterers (StaMPS), a widely used PSI approach, will be introduced.

3.3. The Stanford Method for Persistent Scatterers (StaMPS)

StaMPS, as an open-source PSI method, is widely used in InSAR deformation analysis. In this section, following Hooper et al. (2007), the details of this algorithm will be introduced.

A key step in PSI is the identification of the PSC, which involves finding the pixel with the minimum ϕ_n (as outlined in Eq. 3.23 and Eq. 3.4). To estimate ϕ_n, $\Delta\varphi_{topo}$ and the spatially correlated phase must first be modeled and subtracted from $\Delta\phi$. However, estimating phase noise becomes computationally expensive since the raw phase is wrapped. To address this challenge, a pre-selection of PSC proves helpful. As suggested by Ferretti et al. (2001), the amplitude dispersion D_A can be used as an estimator of phase standard deviation σ_ϕ, when the D_A value is low, as illustrated in Eq. 3.24 and Fig. 3.4.

$$\sigma_\phi \approx D_A = \frac{\sigma_A}{\mu_A}, \tag{3.24}$$

σ_A and μ_A denote the standard deviation and mean of the amplitude of a pixel, respectively. Before StaMPS, most of PSI algorithms started with a small set of good PSC, just like Ferretti et al. (2001) suggested, $D_A < 0.25$ was recommended as the threshold for selecting PSC. But for StaMPS, the algorithm starts with a more relaxed PSC set, which may contain non-PS but hopefully all PS, and then reject those candidates hard become PS. Following this strategy, $D_A < 0.4$ is a more appropriate threshold. This setting can not only ensure that enough PSC are retained but also filter out most of the non-PSC, thereby greatly reducing the computational cost.

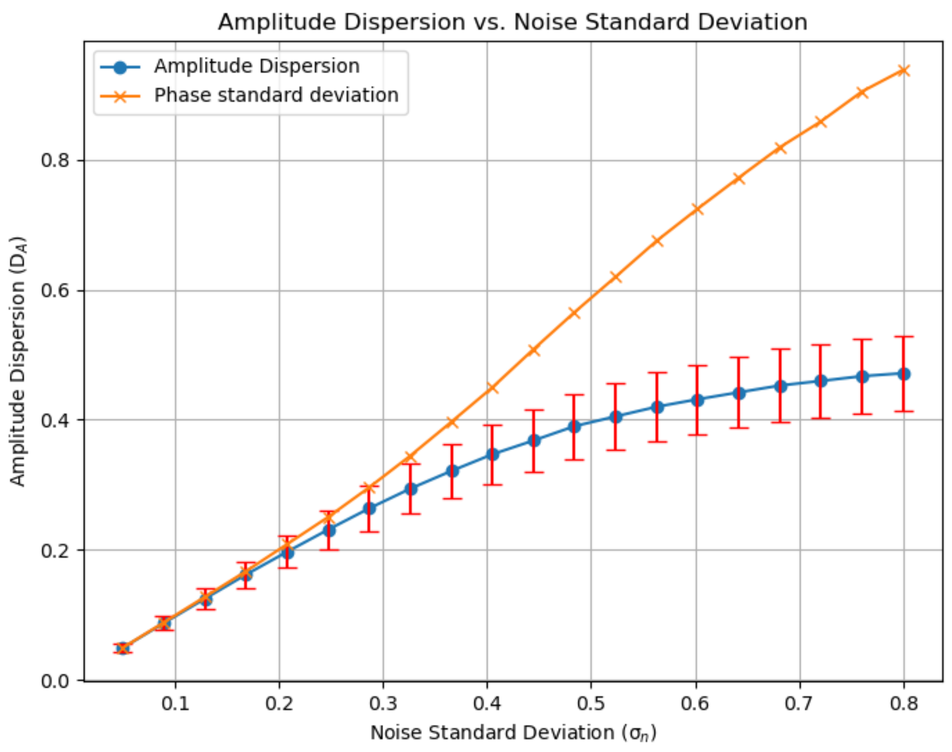


Fig. 3.4 Simulation of amplitude dispersion D_A and phase standard deviation σ_ϕ under different noise levels. The error bar represents the standard deviation of 5000 simulated D_A for each σ_n. This simulation is done according to Hooper et al. (2007).

After PSC selected by D_A, an iterative process involving phase estimation is performed to remove pixels that are unlikely to be PS. After removing φ_{flat}, φ_h and doing the coregistration by pre-processing, Eq.3.4 can be expressed as:

$$\Delta\phi = W\left(\varphi_{defo} + \varphi_{atmo} + \Delta\varphi_{orbit} + \Delta\varphi_{topo} + \varphi_{noise}\right). \tag{3.25}$$

In Eq. 3.25, the first three terms and part of the fourth term are assumed to be spatially correlated and can therefore be removed by a spatial bandpass filter. For the spatially uncorrelated part of the fourth term $\Delta\varphi_{topo}^{su}$, a linear search is required to estimate its periodogram. For StaMPS, the estimation of these terms is performed iteratively, and at the beginning of each iteration, $\Delta\varphi_{topo}^{su}$ is first estimated and eliminated as follows:

According to Eq. 3.8, the $\Delta\varphi_{topo}^{su}$ is linearly related to the spatially uncorrelated part of the look angle error $\Delta\theta^{su}$ via the perpendicular baseline $B_{\perp,x,i}$, so $\Delta\varphi_{topo}^{su}$ becomes:

$$\Delta\varphi_{topo}^{su} = \frac{4\pi}{\lambda} B_{\perp} \Delta\theta^{su}. \tag{3.26}$$

Due to the 2π ambiguity of the phases, the periodogram estimation is nonlinear. In StaMPS, the $\Delta\theta^{su}$ is estimated as following search approach: First, set a maximum DEM error value and convert it into phase by using Eq. 3.26. Then a linear search space is constructed following that phase. Each element in this linear space is used as a potential DEM error to calculate the coherence, and the phase that can obtain the maximum coherence in this space is selected as the rough estimation of $\Delta\theta^{su}$ and then be subtracted from $\Delta\phi$. Finally, a least squares fitting is performed on the remaining phase to obtain the best periodogram estimation of $\Delta\varphi_{topo}^{su}$.

After obtaining the best estimation of $\Delta\varphi_{topo}^{su}$ and subtracting its wrapped phase from $\Delta\phi$, the remaining is:

$$\Delta\phi - W\left(\Delta\varphi_{topo}^{su}\right) = W\left(\varphi_{defo} + \varphi_{atmo} + \Delta\varphi_{orbit} + \Delta\varphi_{topo}^{sc} + \varphi_{noise}\right), \tag{3.27}$$

where $\Delta\varphi_{topo}^{sc}$ denotes the spatially correlated part of DEM error. Since the first four terms in Eq. 3.27 has been assumed to be spatially correlated, an adaptive phase filter has been designed to filter out the spatially correlated phase by using a combination of a fifth-order Butterworth filter and the Goldstein-Werner filter, as described below:

$$G(u,v) = L(u,v) + \beta max[(\left(\frac{H(u,v)}{\overline{H}(u,v)}\right)^{\alpha} - 1),0], \tag{3.28}$$

$$H(u,v) = |Z(u,v)|, \tag{3.29}$$

where $Z(u,v)$ is the smoothed absolute phase value of the 2-D FFT (window size 32 or 64) by convolution with e.g. a 3×3 Gaussian window (Goldstein & Werner, 1998) (StaMPS use 7×7). $L(u,v)$ is the phase after filtering by a 5th order Butterworth filter, with a typical cutoff wavelength of 800 m, α and β are adjustable weighting parameters, typical values being 1 and 0.3, respectively. $\overline{H}(u,v)$ is the median value of $H(u,v)$.

Before filtering, the amplitude of each pixel is weighted by an estimate of the signal-to-noise ratio (SNR) and sampled to a regular grid at intervals of 40 m to 100 m. For the first iteration, the SNR is assumed to be $SNR = \frac{\mu_A}{\sigma_A}$, and in the later iterations, the SNR for pixel x has been set as:

$$SNR = \frac{g_x^2}{2\sigma_n^2}, \tag{3.30}$$

where the signal g_x can be estimated by the noise phase ϕ_{noise} from the last iteration as:

$$\hat{g}_x = \frac{1}{N}\sum_{i=1}^{N} A_{x,i} \cos\phi_{noise,x,i}, \tag{3.31}$$

where $A_{x,i}$ is the amplitude of pixel x in the i-th interferograms, and the noise variance can be estimated as:

$$\hat{\sigma}_n^2 = \frac{1}{2}\left[\frac{\sum_{i=1}^{N} A_{x,i}^2}{N} - \left(\frac{1}{N}\sum_{i=1}^{N} A_{x,i}\cos\phi_{noise,x,i}\right)^2\right]. \tag{3.32}$$

After the above filtering process, the spatially correlated phase ϕ^{sc} is considered to be filtered out. Subtracting ϕ^{sc} from $\Delta\phi - W\left(\Delta\varphi_{topo}^{su}\right)$, there is:

$$\Delta\phi - W\left(\Delta\varphi_{topo}^{su}\right) - \phi^{sc} = W\left(\varphi_{defo}^{su} + \varphi_{atmo}^{su} + \Delta\varphi_{orbit}^{su} + \varphi_{noise}^{su}\right). \tag{3.33}$$

where the superscript su denotes the spatially uncorrelated part of the phase. Assuming that φ_{defo}^{su}, φ_{atmo}^{su}, and $\Delta\varphi_{orbit}^{su}$ are mainly affected by long-wavelength signals, the sum of the first three terms on the right-hand side can be replaced by a small perturbation term δ, therefore, there is:

$$\Delta\phi - W\left(\Delta\varphi_{topo}^{su}\right) - \phi^{sc} = W(\varphi_{noise}^{su} + \delta). \tag{3.34}$$

Then the coherence γ can be calculated by summing those remaining phases along time as:

$$\gamma = \frac{1}{N}\sum_{i=1}^{N} e^{i\left(\Delta\phi - W\left(\Delta\varphi_{topo}^{su}\right) - \phi^{sc}\right)}. \tag{3.35}$$

The calculation of γ is iterative, assume $\delta \approx 0$, the residual phase can be considered as ϕ_{noise} for estimate the SNR in the next iteration until γ convergence. Then, whether pixel x belongs to PS can be determined by γ_x and $D_{A,x}$ with the following approach:

1) Calculation of the probability density function $pdf(\gamma_x)$ by binning and normalizing the calculated γ_x.
2) Assuming that $pdf(\gamma_x)$ consists of two components, which are: probability density function for x belongs to PS $pdf_{PS}(\gamma_x)$, and x doesn't belong to PS $pdf_{NPS}(\gamma_x)$. Then the $pdf(\gamma_x)$ can be written as:
$$pdf(\gamma_x) = \alpha pdf_{PS}(\gamma_x) + (1-\alpha)pdf_{NPS}(\gamma_x), \qquad 0 \le \alpha \le 1. \tag{3.36}$$
3) Using random phases in $[-\pi, \pi]$ to generate the $pdf_{NPS}(\gamma_x)$.
4) For $\gamma_x \le 0.3$, $pdf_{PS}(\gamma) \approx 0$, then the α can be solved by:
$$\int_0^{0.3} pdf(\gamma_x)d\gamma_x = \int_0^{0.3}(1-\alpha)pdf_{NPS}(\gamma_x)d\gamma_x. \tag{3.37}$$
5) Given an acceptable false positive probability q, the γ_x threshold of PS $\gamma_{threshold}$ with different $\widehat{D}_{A,x}$ calculated by ϕ_{noise} can be determined by solving:
$$\frac{\left(1-\alpha(\widehat{D}_{A,x})\right)\int_{\gamma_{threshold}}^{1} pdf_{NPS}(\gamma_x)d\gamma_x}{\int_{\gamma_{threshold}}^{1} pdf(\gamma_x)d\gamma_x} = q. \tag{3.38}$$
6) $\gamma_{threshold}$ is approximately linearly related to $\widehat{D}_{A,x}$, with using the least-square-method to solve the best fit of constant k for $\gamma_{threshold} = k\widehat{D}_{A,x}$, the pixel with $\gamma > k\widehat{D}_A$ are selected as PS pixel.

Once the PS pixels are identified, the PS network they form can be used to perform phase unwrapping via SNAPHU. To minimize the unwrapping error, the previously estimated $W\left(\Delta\varphi_{topo}^{su}\right)$ need to be removed from $\Delta\phi$ before unwrapping. After phase unwrapping, the unwrapped phase still contains spatially correlated components such as $\varphi_{atmo}, \Delta\varphi_{orbit}$ and $\Delta\varphi_{topo}^{sc}$. In StaMPS, these components are filtered out using temporal low-pass and high-pass filtering, respectively. Since these spatially correlated phases also introduce unwrapping errors, the filtering and unwrapping process needs to be repeated until the phase converges. This method ensures that the unwrapped displacement phase is accurately obtained so that the deformation analysis can be completed using Eq. 3.9.

Using the above method, the StaMPS approach can effectively perform deformation analysis when the PS are dense enough with millimeter-level accuracy. However, in rural areas where PS are scarce, insufficient PS leads to increased phase unwrapping errors, which reduces its reliability for large-scale deformation monitoring. To address this limitation, the next section introduces an extension of the StaMPS processing chain that incorporates DS. This extension can jointly process PS and DS, providing more reliable deformation monitoring.

3.4. Joint processing of Distributed and Persistent Scatterers

Unlike PS, which represents deterministic radar targets, DS consists of stochastic targets that follow certain statistical descriptions, which may change from place to place. Since the properties of the scatterer may change over time, and their coherence is usually low, the phase for individual DS is usually not directly usable. However, for some DS, their neighboring pixels may share similar backscattering statistical characteristics. This allows for the enhancement of the SNR of these scatterers through statistical techniques, such as averaging pixels (multilooking). Although the exact displacement of a single DS cannot be measured directly, statistical methods can estimate the average displacement of a group of DS with similar statistical properties by interferometry (Guarnieri & Tebaldini, 2008). This idea led to the development of two parallel DS processing methods: Small Baseline Subsets (SBAS) (Berardino et al., 2002) and SqueeSAR (Ferretti et al., 2011).

For a single pixel, the phase of a certain interferogram $\Delta\phi_1$ can be constructed from two other interferograms $\Delta\phi_2$ and $\Delta\phi_3$. This is also called triangular phase as shown in Eq. 3.39:

$$\Delta\phi_1 - \Delta\phi_3 = (\Delta\phi_1 - \Delta\phi_2) + (\Delta\phi_2 - \Delta\phi_3) \tag{3.39}$$

However, after spatial filtering the interferograms phases no longer satisfy Eq. 3.39. Therefore, statistical methods are usually used to estimate the DS phase from multiple redundant interferograms.

Different from PSI, SBAS selects specific SAR image pairs for interferometry based on the favorable baselines, rather than selecting a master image and interfering with all slave images to obtain the interferogram stack. For SBAS approach, L subsets with M interferograms with favorable baselines (small baselines are preferred) are formed from $N+1$ SAR images $\boldsymbol{\phi} = [\phi_0, \phi_1, \phi_2, \dots, \phi_N]$. The interferograms stack for SBAS follows:

$$\Delta\varphi_j = W^{-1}(\phi_{iE} - \phi_{iS}), \qquad iE > iS, \tag{3.40}$$

Where W^{-1} is the unwrapping operator and iS and iE are certain acquisition index of a subset selected by favorable baselines. The interferogram stack $\boldsymbol{\Delta\varphi}$ become:

$$\boldsymbol{\Delta\varphi}^{\mathbf{T}} = [\Delta\varphi_1, \Delta\varphi_2, \dots, \Delta\varphi_M], \qquad M \geq N - L + 1. \tag{3.41}$$

For $L = 1$, a system with M equations and N unknowns of Eq. 3.40 and Eq. 3.41 can be written as:

$$\boldsymbol{\Delta\varphi} \ = \boldsymbol{C}W^{-1}(\boldsymbol{\phi}) \ , \tag{3.42}$$

where $\boldsymbol{C}$ the adjacency matrix of the connection graph formed by iE and iS, like:

$$\boldsymbol{C} = \begin{bmatrix} 1 & 0 & \dots & -1 & 0 \\ 0 & 1 & & 0 & 1 \\ \vdots & & \ddots & \vdots & \\ 1 & 0 & \dots & 0 & -1 \\ 0 & 0 & & 1 & -1 \end{bmatrix}, with\ C_{j,iE} = 1, C_{j,is} = -1, others = 0. \tag{3.43}$$

The least-squares solution of this system is:

$$\widehat{\boldsymbol{\varphi}} = \boldsymbol{C}^{+}\boldsymbol{\Delta\varphi} \ , \qquad \boldsymbol{C}^{+} = \left(\boldsymbol{C}^{T}\boldsymbol{C}\right)^{-1}\boldsymbol{C}^{T}. \tag{3.44}$$

In some cases, the number of subsets L may be larger than 1. In that situation, the rank of matrix $\boldsymbol{C}$ is $N - L + 1 < N$, meaning the system of equations need be solved using Singular Value Decomposition (SVD).

A key limitation of the SBAS method is that it requires phase unwrapping of the whole interferogram before the least squares solution. This means that phase unwrapping errors can corrupt the DS phase estimate. For scenes with low coherence, such as areas with dense vegetation canopies, the phase unwrapping results may not meet expectations, causing the solution to fail. Therefore, estimating the DS signal before any unwrapping errors occur becomes a more promising approach for DS analysis.

In order to retain the advantage of SBAS in estimating the DS phase from multiple interferograms while avoiding the import of destructive phase unwrapping errors, the SqueeSAR method was introduced. SqueeSAR uses statistical methods to select DSC and estimate their phases before phase unwrapping, this key step is called DS preprocessing. The preprocessed DSCs can be jointly processed as virtual PSC by any PSI method, such as StaMPS. Therefore, in the phase unwrapping process of the joint processing, the addition of preprocessed DSC can provide higher pixel density, significantly reducing the unwrapping error and improving the overall accuracy.

DS consists of many small scatterers of similar size within a resolution cell, and the phases of these scatterers can be described by a complex circular Gaussian distribution. When a sufficient number of DS with similar statistical characteristics are identified, their average effective phase can be estimated by phase triangulation. This estimation is performed directly from the wrapped phase without the need for an unwrapping step. The idea above forms the core of the SqueeSAR algorithm. In SqueeSAR, DS preprocessing is performed by the following steps:

1) Grouping of Statistical Homogeneous Pixels (SHP).
2) Estimation of the covariance matrix.
3) Using the Phase Triangulation Algorithm (PTA) to estimate the phase of DSC.
4) Calculation of a quality number (coherence-like index) for the DSC.

The first step in DS preprocessing is the identification of SHP. Since DS and PS are processed jointly, any filtering process must preserve information related to individual PS. Specifically, PS with high coherence should not be averaged with neighboring pixels that may have low

SNR. Therefore, the spatial filter employed must be spatially adaptive, ensuring that only SHP are averaged while keeping the information related to point targets (PS) intact. In the SqueeSAR, SHP selection is performed using an algorithm called DespecKS. For the registered stack of N+1 SAR images, the signal vector corresponding to a particular pixel x in the stack $d(x)$ is given by:

$$d(x) = [z_0(x), z_1(x), \dots, z_N(x)]. \tag{3.45}$$

If x is a PS, $d(x)$ is a deterministic vector, while if x is a DS, then $d(x)$ is a complex random vector. Given two pixels vector $d(x_1)$ and $d(x_2)$, if the null hypothesis that two pixels x_1 and x_2 come from the same probability distribution function cannot be rejected, then the two image pixels x_1 and x_2 are defined as statistically homogeneous. In DespecKS, this hypothesis has been tested by the two-sample Kolmogorov-Smirnov (KS) test (Stephens, 1970). More specifically, the sorted amplitude vector

$$As = [As_0, As_1, \dots, As_N\ ] = sort[A_0, A_1, \dots, A_N] \tag{3.46}$$

of $d(x)$ can be easily converted into an unbiased estimator $S_N(A)$ of the cumulative distribution function (cdf) by:

$$S_N(A) = \begin{cases} 0 & A < As_1 \\ \frac{k}{N} & As_k < A < As_{k+1} \\ 1 & A > As_{N+1} \end{cases}. \tag{3.47}$$

The two-sample KS test measure the maximum value of the absolute difference D_N between the cdf of two pixels x_1 and x_2 as:

$$D_N = \sqrt{\frac{N}{2}} \sup_{A \epsilon R} \left| S_N^{x_1}(A) - S_N^{x_2}(A) \right|. \tag{3.48}$$

The distribution of D_N can be approximated by the KS distribution (Stephens, 1970), as:

$$P(D_N \le t) = H(t) = 1 - 2 \sum_{n=1}^{\infty} -1^{n-1} e^{-2n^2 t^2}. \tag{3.49}$$

Given that the significance level α, if D_N smaller than a threshold c depends on α, the null hypothesis is accepted, which means two pixels x_1 and x_2 are statistically homogeneous and belong to the same group of SHP. The DespecKS algorithm works as follows:

1) For each pixel x in the interferogram stack, define an estimation window that is centered on x.
2) Perform the two-sample KS test mentioned above on each pixel in this window with x, and discard the pixels rejected by the test.
3) Discard those pixels that are not connected with x.
4) Keep the remaining pixels as a group of SHP Ω.

After separating pixels into different SHP groups, the sample covariance matrix of pixel x can be estimated by its SHP group Ω as:

$$C(x) = E[d(x) * d^H(x)] \approx \frac{1}{|\Omega|} \sum_{x \in \Omega} d(x) * d^H(x) = \hat{C}. \tag{3.50}$$

where $*^H$ denotes Hermitian conjugation. After normalization, $\hat{C}$ has been converted to an estimated coherence matrix $\hat{\Gamma}$, the absolute values of the off-diagonal elements of $\hat{\Gamma}$ are an estimate of the coherence values γ_{jk} for all possible interferograms of the data stack, as:

$$\hat{\Gamma} = \{\gamma_{jk} e^{i\Delta\phi_{jk}}\}, \tag{3.51}$$

where $\Delta\phi_{jk}$ denotes the interferogram phase averaged over Ω generated by SAR image j and k.

When the coherence matrix $\hat{\Gamma}$ is estimated, the phase of the DS can be estimated with PTA. For PS, the phase values of the coherence matrix $\hat{\Gamma}$ are redundant. Hence the following equation holds:

$$\angle(\hat{\Gamma}_{nj}) = \angle(\hat{\Gamma}_{nm}\hat{\Gamma}_{jm}^{*}) = \theta_n - \theta_j, \qquad n, j, m = 1,2 \ldots., N+1. \tag{3.52}$$

where $\angle$ denotes the argument of a complex number. This is equivalent to phase triangulation shown in Eq. 3.39. For DS, this is no longer true. To estimate the phase of DS, the coherence matrix Γ can be decomposed into:

$$\Gamma = \Theta \Upsilon \Theta^{H}, \tag{3.53}$$

where Υ is an $N+1 \times N+1$ symmetric real-value matrix whose elements correspond to the coherence values of all the interferograms, and $\Theta = \mathrm{diag}(e^{i\theta})$ is an $N+1 \times N+1$ diagonal matrix, containing the phase information. Assuming SHP in the same group Ω share the same θ, the pdf of this group SHP can be expressed as:

$$p(d_{\Omega}|\theta) \propto \prod_{x\in\Omega} e^{-d_x^H \Theta \Upsilon^{-1} \Theta^H d_x} = e^{-trace(\Theta \Upsilon^{-1} \Theta^H \hat{\Gamma})}. \tag{3.54}$$

Hence, the maximum likelihood (ML) estimation of θ is obtained by maximizing $p(d_{\Omega}|\theta)$ or minimizing the absolute value of its logarithm, this turns the estimation of θ into a nonlinear optimization problem. As there are the phase differences in $\hat{\Gamma}$, phase angles can only be estimated up to an arbitrary additive constant. Without loss of generality, the phase angle of the first interferogram is set 0 ($\vartheta_1 = 0$). The optimal estimated phase angle $\vartheta = [0, \vartheta_2, \vartheta_3, \ldots \vartheta_{N+1}]$ is then given as:

$$\hat{\vartheta} = \underset{\vartheta}{argmax} \left\{\Lambda^{H} \left(|\hat{\Gamma}|^{-1} \odot \hat{\Gamma}\right) \Lambda\right\}, \tag{3.55}$$

where $\Lambda = e^{i\vartheta}$ is an N+1-dimensional vector, and $\odot$ represents the Hadamard product. The optimization process can be implemented by an iterative optimizer, such as Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm.

Once the optimal solution is obtained, the quality number γ_{PTA} of the "goodness fit of the estimated phase" of DS candidates (similar to γ for PS) can be calculated by:

$$\gamma_{PTA} = \frac{1}{N^2 + N} \sum_{n=0}^{N} \sum_{k \neq n}^{N} e^{i\phi_{nk}} e^{-i(\vartheta_n - \vartheta_k)}. \tag{3.56}$$

After calculating γ_{PTA}, good DSC can be selected by applying a threshold to γ_{PTA}, similar to the selection process for PS based on γ. DSC with γ_{PTA} higher than that threshold can be considered to be DS. Once DS pixels are identified, their original phase in the SAR image stack is replaced with the estimated optimal phase $e^{i\vartheta}$. These DS can then be treated like PS, allowing for the joint processing of DS and PS using any PS processing chain.

DS preprocessing estimates the optimal phase values by determining the $N+1$ phase values that best fit the sample coherence matrix before phase unwrapping. This approach utilizes all possible interferograms to estimate the best phase value, thereby avoiding the complex phase unwrapping required in the SBAS method. As an extension of the StaMPS method, DS preprocessing significantly enhances the ability to monitor rural area deformation. However, this method relies on analyzing the neighborhood of each pixel, which becomes a bottleneck that limits the computational efficiency of the method. In Chapter 6 of this thesis, a method for identifying DSC using deep neural networks will be introduced to improve the processing efficiency of DS preprocessing. By adopting deep learning, the neural network can infer DSC using only a pair of polarimetric SAR images with low and high coherence. This innovation can highly accurately identify masks that meet the DSC criteria, allowing only pixels covered by the mask to be processed by the preprocessing step, thereby improving the whole processing speed.

4. The basic framework of deep learning

This chapter will introduce the basics of deep learning related to this thesis. The foundational elements of deep learning, including the multiple layer perceptron model, convolutional networks, sequence modeling and recurrent networks, loss functions, optimizers, and gradient-based learning algorithms are detailed herein. In this section, only the necessary deep learning background knowledge to understand this thesis will be introduced. For other details, please refer to the textbook: *Deep Learning* (Goodfellow et al., 2016) *and Artificial Intelligence A Modern Approach* (Russell & Norvig, 2021)

4.1. The background of deep learning

In the past decade, modern artificial intelligence technology has made rapid progress, largely due to the development of high-performance parallel computing. These technologies have been widely used in various fields, such as computer vision (Goodfellow et al., 2016), speech recognition (Hinton et al., 2012), natural language processing (Brown et al., 2020), and biopharmaceuticals (Jumper et al., 2021). From the perspective of knowledge representation, knowledge acquisition or learning can usually be achieved in two main forms: induction and deduction. Induction is the summary of general rules from a large number of observations, while deduction is logical reasoning based on known rules to derive new knowledge. In the early days of artificial intelligence, computing power was limited. The typical learning method was to use physical, mathematical, or statistical principles to derive "models" with clear mathematical forms and then use machines to determine the parameters of these models based on observed results. This process is usually called "machine learning." However, due to computing requirements and engineering challenges, the number of model parameters that can be learned is usually limited. Therefore, linear systems are usually used to approximate the model to simplify calculations. Although kernel methods can give machine learning models certain nonlinear fitting capabilities, their performance usually depends on the appropriateness of the input feature design. This dependency means that machine learning models often rely on the expertise of domain experts and struggle with tasks that are relatively intuitive to humans but difficult to express in mathematical terms.

The real world often consists of complex nonlinear systems, but human intuition can easily and accurately solve such complex nonlinear problems. Therefore, more "intelligent" learning algorithms should enable machines to learn nonlinear patterns that are difficult to describe directly from experience, similar to human learning, and should avoid the need for complex assumptions. Fortunately, with the advancement of Graphics Processing Unit (GPU) parallel computing technology, it has been discovered that when Artificial Neural Networks (ANN) possess sufficient depth and parameters, they can directly learn complex nonlinear knowledge representations from observed data without the need for manual feature engineering (Lecun et al., 2015). This capability efficiently allows ANN become deeper and deeper, then the deep artificial neural networks, abbreviated as DNN, has been put forward to effectively utilize the increased computing power brought about by computer hardware innovations and the increased data availability brought about by advances in sensor technology, thereby achieving better performance than models manually designed by human experts—in other words, becoming more "intelligent." This unique property has gradually given rise to a new branch of traditional machine learning, i.e. deep learning.

Since artificial feature engineering "guidance" is typically not introduced in the learning process, ANN uses several common simple modules, also known as neurons, to stack and combine, thereby automatically identifying patterns in the data. Neurons (McCulloch & Pitts, 1943) are usually composed of simple, trainable weight w_i and bias b, as shown in Fig. 4.1. The data propagation within a neuron can be expressed as the inner product between the input data x_i and the weight w_i then followed by a nonlinear transformation performed by the activation function σ, yielding the output y, as shown in Eq. 4.1.

$$y = \sigma\left(\sum_i w_i x_i + b\right). \tag{4.1}$$

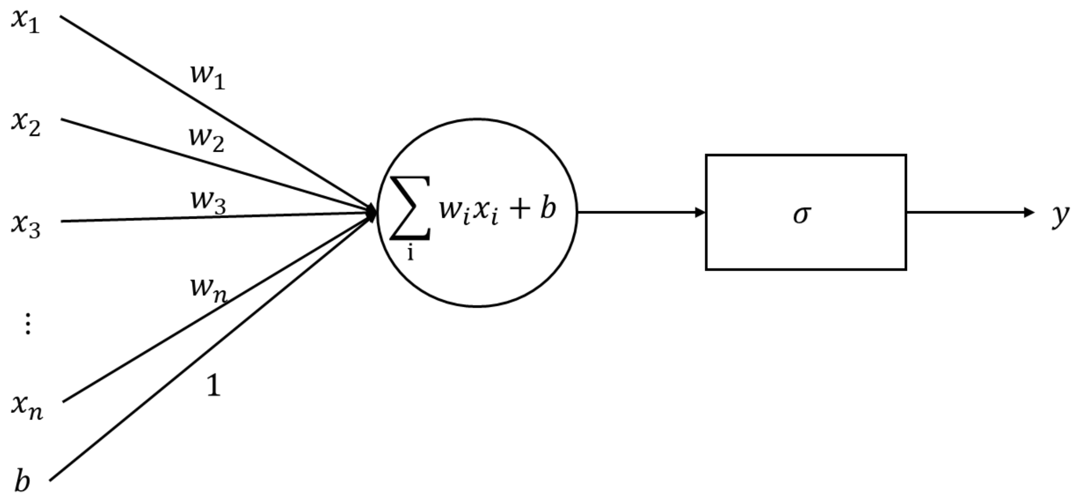


Fig. 4.1 McCulloch–Pitts neuron model

Therefore, neurons can also be regarded as units of nonlinear transformation. Although such nonlinear transformations are usually simple, such as hyperbolic tangent or sigmoid function, stacking multiple layers of neurons will produce a composite of multiple nonlinear functions, giving it powerful nonlinear fitting capabilities. The Kolmogorov–Arnold representation theorem, also known as the Universal Approximation Theorem (Tikhomirov, 1991), demonstrates that a composite of more than three layers of neurons can fit any multivariate function. This theorem is the theoretical basis for the effectiveness of deep learning methods. However, the universal approximation theorem only shows that there is a set of suitable neurons that can fit the target function but does not provide a method to find these neurons. In deep learning, the number of neurons usually reaches millions, so it is impractical to find suitable neurons through exhaustive search. Fortunately, the BP algorithm (Rumelhart et al., 1986) enables the updating of neurons based on the gradient of the loss function, which measures the discrepancy between the neural network's output and the observed reference (in deep learning, it is called the ground truth). This approach transforms the training of the neural network into an optimization problem that can be effectively solved.

To enable machines to recognize patterns in data, architecture, i.e., the organizational structure of neurons, is crucial in deep learning. Neurons in DNN are organized into different layers, such as convolutional layers or fully connected layers, based on their functions. As the basic components of architecture, these common layers and architectures are introduced in the subsequent parts of this chapter. As the basic components of architecture, these common layers and architectures are introduced in the subsequent parts of this chapter. In DNN, different layers of the network can learn and combine features at different levels of abstraction. For example, in the context of visual features, the shallower layers in a neural network usually learn basic features such as color and texture. In contrast, the deeper layers integrate these primary features into higher-level, conceptual semantic features that are comprehensible to humans, such as hair and appearance (Zeiler & Fergus, 2014). Importantly, these features in DNN are learned entirely automatically from data, enabling the outputs of different layers in DNN to serve as machine-extracted features equivalent to those manually designed by human experts.

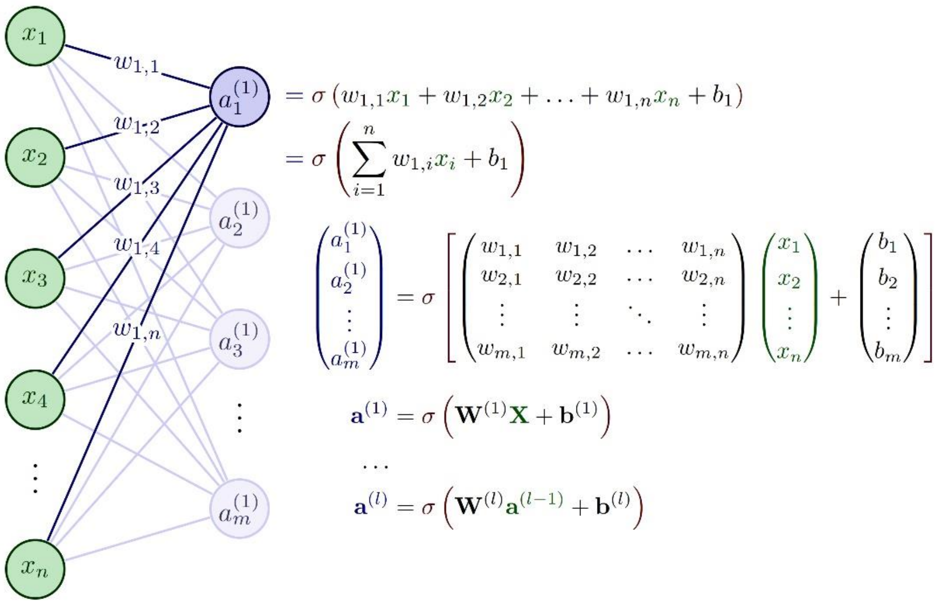
4.2. Perceptron model and multiple layer perceptron

The perceptron model is the simplest ANN model, invented by Warren McCulloch and Walter Pitts in 1943 alongside the concept of neurons (McCulloch & Pitts, 1943). This model is an algorithm inspired by biology and neuroscience, designed to simulate synapses in the nervous system. In 1957, Frank Rosenblatt implemented the perceptron model in hardware at the Cornell Aeronautical Laboratory, turning it into a physical machine (Rosenblatt, 1960). The original perceptron model consists of a single neuron, with its mathematical expression given by Eq. 4.1. To simulate the activation and inhibition states of biological neurons, the original perceptron uses the Heaviside step function σ_H, as shown in Eq.4.2, as the activation function. This makes the perceptron output only two states, 0 or 1, thereby functioning solely as a discriminator.

$$\sigma_H(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases} . \tag{4.2}$$

Although the perceptron model looked promising at first, it soon became apparent that it could not be trained to recognize many types of patterns. A single-layer perceptron could only learn linearly separable patterns, meaning it could only use a straight line to separate data points into two different groups. This limitation prevented it from handling nonlinear problems, such as the XOR problem. Fortunately, the advent of Multi-Layer Perceptrons (MLP, also known as feedforward neural networks or deep feedforward networks) solved this problem. This development marked the beginning of deep learning.

Formally, MLP is a layered stacking of several perceptrons, typically consisting of more than three layers. Unlike the raw perceptron model, the layers of an MLP have a notion of width, i.e. they consist of multiple neurons in parallel. This allows data propagation to be represented in matrix form for parallel computation. Let the input data be $X = \{x_1, x_2, \dots, x_n\}$, $w_i = \{w_{i,1}, w_{i,2}, \dots, w_{i,n}, b_i\}$ is the weights and bias of the i-th neuron, so $W^{(l)} = \{w_1, w_2, \dots, w_m\} = \begin{pmatrix} w_{1,1} & \cdots & w_{1,n} \\ \vdots & \ddots & \vdots \\ w_{m,1} & \cdots & w_{m,n} \end{pmatrix}$ represents the *m* neurons in the layer *l*, $a_i^{(l)}$ is the output of the i-th neuron at layer *l*, so $a^{(l)} = \{a_1^{(l)}, a_2^{(l)}, \dots, a_m^{(l)}\}$ represents the output vector of layer *l*. Let σ denotes the activation function, then the data propagation is shown in Fig. 4.2.


x_1
x_2
x_3
x_4
x_n
$w_{1,1}$
$w_{1,2}$
$w_{1,3}$
$w_{1,4}$
$w_{1,n}$
$a_1^{(1)}$
$a_2^{(1)}$
$a_3^{(1)}$
$a_m^{(1)}$
$= \sigma (w_{1,1}x_1 + w_{1,2}x_2 + \ldots + w_{1,n}x_n + b_1)$
$= \sigma \left(\sum_{i=1}^{n} w_{1,i}x_i + b_1 \right)$
$\begin{pmatrix} a_1^{(1)} \\ a_2^{(1)} \\ \vdots \\ a_m^{(1)} \end{pmatrix} = \sigma \left[\begin{pmatrix} w_{1,1} & w_{1,2} & \ldots & w_{1,n} \\ w_{2,1} & w_{2,2} & \ldots & w_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m,1} & w_{m,2} & \ldots & w_{m,n} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix} \right]$
$\mathbf{a}^{(1)} = \sigma \left(\mathbf{W}^{(1)} \mathbf{X} + \mathbf{b}^{(1)} \right)$
$\ldots$
$\mathbf{a}^{(l)} = \sigma \left(\mathbf{W}^{(l)} \mathbf{a}^{(l-1)} + \mathbf{b}^{(l)} \right)$
Fig. 4.2 Data propagation for MLP

In order to keep the features and their gradients propagating normally, the Heaviside step function is no longer suitable as an activation function because it only outputs 0/1. Instead, continuous or nearly continuous nonlinear functions such as hyperbolic tangent (tanh), Sigmoid, and Rectified Linear Unit (ReLU) are used as activation functions to keep the features and their gradients propagating. The sigmoid function is defined as:

$$Sigmoid(x) = \frac{1}{1 + e^{-x}}, \tag{4.3}$$

which is a monotonically increasing nonlinear mapping from $\mathrm{R} \rightarrow (0,1)$ and serves as an excellent alternative to the Heaviside step function, as shown in Fig.4.3. The hyperbolic tangent function is defined as:

$$tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}, \tag{4.4}$$

providing a monotonically increasing nonlinear mapping from $\mathrm{R} \rightarrow (-1,1)$. While it exhibits effective nonlinear mapping properties, its gradient is significant only when the input is near 0, and it becomes nearly zero for very large or small inputs, as shown in Fig.4.3, potentially leading to gradient explosion or vanishing, and consequently, training failure. Therefore, a commonly used activation function is the ReLU, which is simple in form and easy to compute, defined as:

$$ReLU(x) = \begin{cases} x & x \geq 0 \\ 0 & x < 0 \end{cases}. \tag{4.5}$$

ReLU introduces nonlinearity by suppressing inputs less than 0 and is linear everywhere except at 0. This simplicity in derivative computation helps make computation faster in neural networks using ReLU compared to other activation functions.

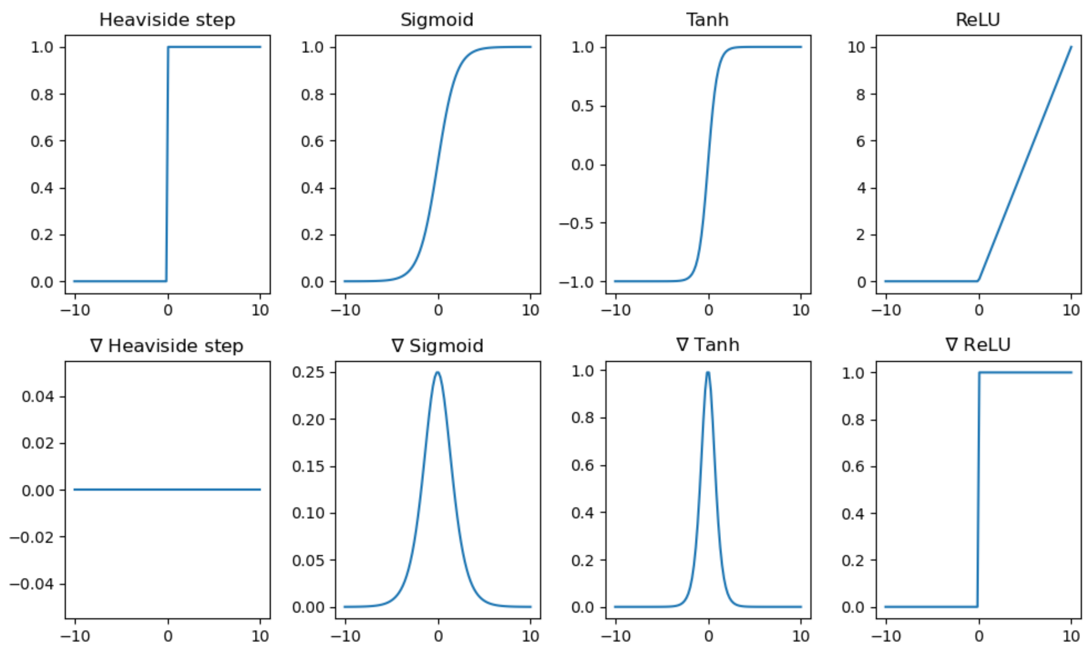


Fig. 4.3 Common activation functions and their gradients

The first layer of an MLP is called the input layer, which consists of neurons that directly multiply the input data. After the data passes through the input layer, it passes through multiple hidden layers, each of which consists of multiple neurons that take the outputs of all neurons in the previous layer as input. Therefore, this structure is also called a fully connected layer or a fully connected network. The output of the hidden layer represents the nonlinear hierarchical features automatically extracted by the network. Shallow features are sequentially combined through deeper hidden layers to form higher-level features. This ability gives MLP significant nonlinear fitting capabilities, making it more practical than a single-layer perceptron. Written in matrix form and include bias $b^{(l)}$ into weights matrix $W^{(l)}$, the output $Y = \{y_1, y_2, \dots, y_k\}$ of an L-layer MLP can be expressed as a composite:

$$Y = \{y_1, y_2, \dots, y_k\} = W^{(L)}(\sigma^{(L)}(W^{(L-1)} \dots \sigma^{(2)}(W^{(2)}\sigma^{(1)}(W^{(1)}X)) \dots) \tag{4.6}$$

where $\sigma^{(l)}$ is the activation function of the l-th layer. The diagram of MLP is shown in Fig. 4.4.

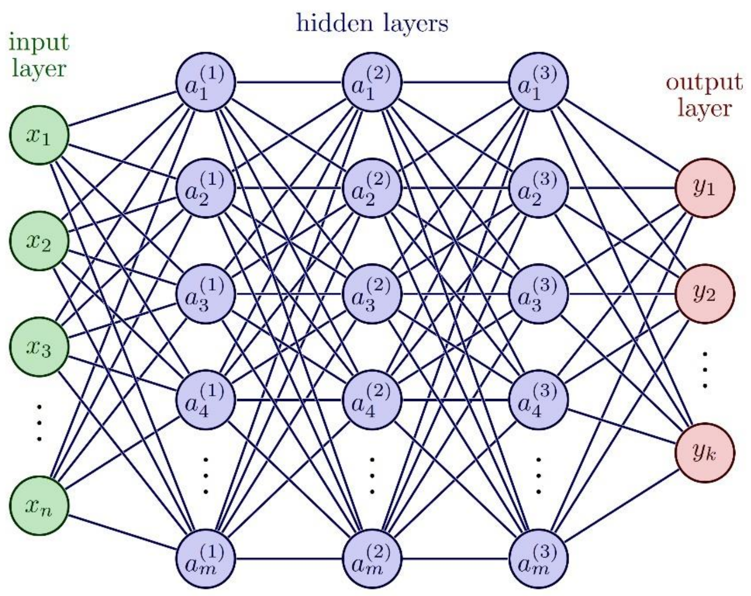


Fig. 4.4 The diagram of MLP

4.3. Loss Function, Gradient-Based Learning and Optimizers

Eq. 4.6 indicates that the output Y of a DNN f_{DNN} is determined by the input X according to a set of trainable weight parameters W, as shown in Eq. 4.7.

$$Y = \{y_1, y_2, \dots, y_k\} = f_{DNN}(X; W). \tag{4.7}$$

However, finding a suitable set of weight parameters W (also called training) to ensure that the output Y is close enough to the ground truth is a challenge. In the early days of machine learning, the total number of parameters of shallow networks that needed to be learned was not large, and the weight parameters W could be determined using search-based algorithms or evolutionary algorithms (such as genetic algorithms, particle swarm optimization, simulated annealing, etc.). However, as networks became deeper and larger, the number of parameters required for training increased significantly, making these early algorithms inefficient or computationally impractical. Fortunately, despite the BP algorithm (LeCun, 1988) being widely misunderstood in the 1990s, its amazing performance in 2012 with AlexNet (Krizhevsky et al., 2012) in the field of computer vision demonstrated its potential for training DNN. AlexNet achieved significantly higher accuracy than traditional machine learning methods and shallow neural networks by combining the backpropagation algorithm with GPU parallel computing to train very deep (at the time) convolutional neural networks. This milestone marked the rise of modern deep learning and the gradual abandonment of manual feature design, and led to the acceptance of backpropagation as an effective algorithm for training DNN.

Before introducing the BP algorithm in detail, it is essential to first understand the concept of the loss function. The loss function, also known as the cost function, is a binary function used to measure the difference or distance between the network output Y and the ground truth $GT = \{gt_1, gt_2, \dots, gt_k\}$, denoted as $Loss(Y, GT)$. It should be emphasized that the term ground truth has different meanings in the fields of machine learning and remote sensing or geodesy. In this thesis, the term ground truth represents factual knowledge relative to a specific problem, that is, observations used as references, or standard answers recognized by humans, rather than the location coordinates. In most geodetic scenarios, the output of a model (whether deep learning or otherwise) is typically a numerical value with a specific unit. For those values, the Euclidean distance is commonly used to measure their differences, which is also known as the Root Mean Square Error (RMSE):

$$RMSE(Y, GT) = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(Y - GT)^2}, \tag{4.8}$$

where N is the number of samples. Therefore, when DNN needs to generate or predict some numerical values based on the input, that is, when performing numerical regression tasks, RMSE is usually used as the loss function. In this case, training the network can be regarded as an optimization task to find a set of weights W that minimize the loss function. This can be expressed as:

$$Loss_{RMSE} = \underset{W}{\operatorname{argmin}}\, RMSE(f_{DNN}(X; W), GT) = \underset{W}{\operatorname{argmin}} \sqrt{\frac{1}{N}\sum_{n=1}^{N}(y_n(x_n; W) - gt_n)^2}. \tag{4.9}$$

Different from numerical regression tasks, another important application of models in remote sensing and geodesy is to classify or identify pixels in remote sensing images, such as ground object recognition and mask generation commonly used in SAR and InSAR applications. When dealing with classification problems, pixels are labeled by category and then One-Hot encoded, which is equivalent to converting the label into a Multinoulli distribution $P(gt)$ (also called categorical distribution) (Robert, 2014) as the target for learning. This means for K categories, one pixel can be labeled and encoded as a K- dimension discrete probability distribution as the target, and in this distribution, only the probability of class c_i with correct label is 1, otherwise it is 0. as shown in Eq. 4.10. An example of one-hot encoding as shown in Fig.4.5.

$$P(gt) = p(c_i = gt) = \begin{cases} 1, & i = gt \\ 0, & i \neq gt \end{cases}. \tag{4.10}$$

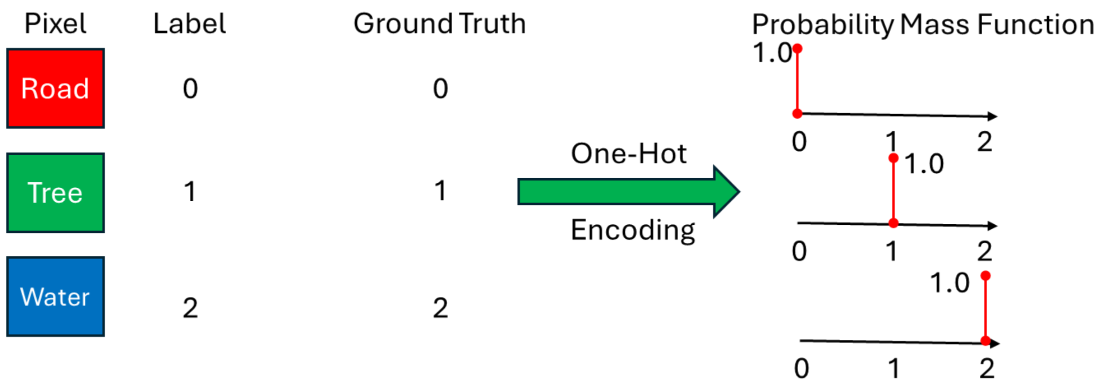


Fig. 4.5 An example of One-hot encoding. In this example, there are three categories of pixels: road, tree, and water, which are labeled 0, 1, and 2, respectively. After One-hot encoding, the target of a pixel that belongs to the road, tree, or water is $\{1,0,0\}, \{0,1,0\}, \{0,0,1\}$ respectively.

.

In this case, the output of a DNN model should also be a discrete probability distribution, and the purpose of training is to let that distribution be as close to the target distribution as possible. For classification tasks, the total number of categories K is knowable. Therefore, the output layer can use K neurons to let the network output K values, and then use the SoftMax function to convert them into probabilities, as shown in Eq. 4.11. This allows the output of DNN to be represented as a discrete probability distribution $Q(y_n)$.

$$Q(y_n) = SoftMax(y_n) = \frac{e^{y_{n,k}}}{\sum_{k=1}^{K} e^{y_{n,k}}}. \tag{4.11}$$

where y_n is the output vector with K-dimension of the network corresponding to the input x_n, $y_{n,k}$ is the value of its k-th component. Then, the loss function needs to measure the similarity between the target distribution $P(gt_n)$ and the DNN output distribution $Q(y_n)$. In statistics, Kullback-Leibler (KL) divergence quantifies the amount of information lost when using probability distribution $Q(y_k)$ to approximate the probability distribution $P(gt_n)$, as shown in Eq. 4.12.

$$\begin{aligned} D_{KL}(P||Q) &= \sum_{k=1}^{K} P(gt_n) \log \frac{P(gt_n)}{Q(y_n)} \\ &= \sum_{k=1}^{K} P(gt_n) \log P(gt_n) - \sum_{k=1}^{K} P(gt_n) \log Q(y_n). \end{aligned} \tag{4.12}$$

The first term in Eq. 4.12 is the entropy of distribution $P(gt_n)$, which represents the average number of bits needed to encode using the optimal code for $P(gt_n)$. Since $P(gt_n)$ is encoded from the ground truth gt_n, this makes this entropy is a constant and can be ignored in the optimization process. The remaining part is the cross entropy H_{CE} between $P(gt_n)$ and $Q(y_n)$, denoted as:

$$H_{CE}(P||Q) = -\sum_{n=1}^{N} P(gt_n) \log Q(y_n). \tag{4.13}$$

By minimizing the cross-entropy $H_{CE}(P||Q)$ as the loss function, as shown in Eq. 4.14, the network can obtain an optimal set of weights W to output a discrete probability distribution $Q(y_n)$ closest to the encoded ground truth Multinoulli distribution $P(gt_n)$. The category with the maximum probability in $Q(y_n)$ is then considered as the classification result of the network.

$$\begin{aligned} Loss_{CE} &= \operatorname*{argmin}_{W} H_{CE}(P(GT)||Q(f_{DNN}(X;W))) \\ &= -\operatorname*{argmin}_{W} \frac{1}{N} \sum_{n=1}^{N} \sum_{k=1}^{K} P(gt_n) \log \frac{e^{f_{DNN}(x_n;W)_k}}{\sum_{k=1}^{K} e^{f_{DNN}(x_n;W)_k}}. \end{aligned} \tag{4.14}$$

In addition to the two commonly used loss functions mentioned above, for more complex tasks, such as fitting unknown distributions, the maximum likelihood method can be used to derive the general form of the loss function. In this case, the loss function can be simplified to the negative log-likelihood, as shown in Eq. 4.15.

$$Loss = -\operatorname*{argmin}_{W} \mathbb{E}_{GT} \log L_{DNN}(f_{DNN}(X;W)), \tag{4.15}$$

where $\mathbb{E}_{GT}$ is the expectation of the distribution from GT, and L_{DNN} is the likelihood of the output distribution of the DNN. In fact, RMSE and cross-entropy H_{CE} can be regarded as special cases of using maximum likelihood to construct loss functions for Gaussian distribution and Multinoulli distribution, respectively. In Chapter 5 of this thesis, the method of using maximum likelihood to derive loss functions to train Gaussian mixture models for fitting ZWD distribution will be introduced in detail.

Once the loss function is constructed, training the network becomes an unconstrained optimization problem. However, since the network consists of multiple layers of nonlinear mappings, this problem becomes an unconstrained nonlinear optimization problem, which is difficult to solve. Initially, it was widely believed that simple gradient descent would get stuck in a bad local minimum, where any small change in the weight changes would not improve the model. However, large networks rarely encounter major problems with poor local minima. Regardless of the initial conditions, the system almost always reaches a solution of similar quality. Recent theoretical and empirical results show that local minima are usually not a serious problem. Instead, the landscape is usually crowded with many saddle points with zero gradient, where the surface bends upward in most dimensions and downward in some other dimensions (Dauphin et al., 2014). Analysis indicates that there are only a few saddle points with a large number of downward-bending directions, but the loss function values at almost all saddle points are very similar (Choromanska et al., 2015). Therefore, it does not matter significantly which saddle point the algorithm gets stuck on. This insight makes it feasible to use the backpropagation algorithm combined with gradient descent and its variants to find W that minimizes the loss function and completes the training of the network.

The BP algorithm addresses a crucial question: how to obtain the gradient of each neuron $\nabla_{w_{i,j}} = \frac{\partial Loss}{\partial w_{i,j}}$, and how to pass error information (i.e. gradient) from the loss function back to the network layer by layer so that these weights can be updated using gradient descent. This process is essentially equivalent to the chain rule in calculus. Formally, training a network involves three iterative steps until the loss converges:

1. **Forward Propagation**: Input X into the network f_{DNN} to obtain the output Y (as shown in Eq. 4.6).
2. **Loss Calculation**: Compute the loss $Loss(Y, GT)$ based on loss function with the output Y and the ground truth GT.
3. **Backward Propagation and Weight Update**: Calculate the gradient of each neuron $\nabla w_{i,j}$, and update each weight using gradient descent or its variants:

$$w_{i,j} \leftarrow w_{i,j} - f(\nabla_{w_{i,j}}).$$

According to Eq. 4.6, the network is composed of multiple layers of neurons. By applying the chain rule, given that $z^{(l)} = W^{(l)}a^{(l)}, and\ a^{(l)} = \sigma^{(l)}(z^{(l)})$, the derivative of the loss in term of the input X can be written in matrix form as:

$$\begin{aligned}\frac{dLoss}{dX} &= \frac{dLoss}{dY}\frac{dY}{da^{(L)}}\frac{da^{(L)}}{dz^{(L)}}\frac{dz^{(L)}}{da^{(L-1)}}\frac{da^{(L-1)}}{dz^{(L-1)}}\frac{dz^{(L-1)}}{da^{(L-2)}}\cdots\frac{da^{(2)}}{dz^{(2)}}\frac{dz^{(2)}}{da^{(1)}}\frac{da^{(1)}}{dz^{(1)}}\frac{dz^{(1)}}{dX^{(1)}} \\ &= \frac{dLoss}{dY}W^{(L)}a^{(L-1)'}W^{(L-1)}\ldots a^{(1)'}W^{(1)},\end{aligned} \tag{4.16}$$

where $\frac{da^{(l)}}{dz^{(l)}} = a^{(l)'}$ is a diagonal matrix of the derivatives of the activation functions, and $\frac{dz^{(l)}}{da^{(l-1)}} = \frac{d(W^{(l)}a^{(l-1)})}{da^{(l-1)}} = W^{(l)}$ is the matrices of weights. The gradient ∇_X of the $Loss$ in terms of the input X is the transpose of derivative, so in matrix form, it can be written as:

$$\nabla_X = W^{(1)^{\intercal}}a^{(1)'}W^{(2)^{\intercal}}a^{(2)'}\ldots W^{(L)^{\intercal}}\nabla_Y. \tag{4.17}$$

where symbol $\intercal$ denotes matrix transpose, and $'$ denotes matrix derivative. Introducing the auxiliary $\delta^{(l)}$, which represents the error of layer l, as:

$$\delta^{(l)} = a^{(l)'}W^{(l+1)^{\intercal}}a^{(l+1)'}\ldots W^{(L)^{\intercal}}\nabla_Y. \tag{4.18}$$

And $\delta^{(l)}$ can easily be recursively computed as:

$$\delta^{(l)} = a^{(l)'}W^{(l+1)^{\intercal}}\delta^{(l+1)}. \tag{4.19}$$

The gradient of the weights in layer l is then:

$$\nabla_{W^{(l)}} = \delta^{(l)}\left(a^{(l-1)}\right)^{\mathsf{T}}. \tag{4.20}$$

Since the activated output $a^{(l)}$ of each neuron is known after forward propagation, and the derivatives of the activation functions $a^{(l)'}$ can be known priorly (the derivatives of common activation functions will be described below), the gradient of each neuron can be easily iterated by using Eq. 4.19 and Eq. 4.20 after computing the gradient in terms of the output ∇_Y. For each gradient computing, only three matrix multiplications are required, which greatly saves memory usage and improves the efficiency of this algorithm. In fact, for the commonly used RMSE loss and cross-entropy loss, the gradient of the loss function with respect to the output is straightforward. For RMSE loss, its gradient is:

$$\nabla_Y RMSE = \frac{2}{N}(Y - GT). \tag{4.21}$$

And for cross-entropy loss with one-hot encoding, due to $P(c_i)$ is 1 only for $i = GT$ otherwise is 0, denote $s_i = SoftMax(y_i)$, so the cross-entropy loss become:

$$H_{CE}(P||Q) = -\sum_{i=1}^{K} P(c_i)\log Q(y_i) = -\log Q(y_{GT}) = -\log s_{GT}\,. \tag{4.22}$$

And the gradient is:

$$\nabla_Y CE = -\frac{1}{s_{GT}}\frac{\partial s_{GT}}{\partial Y}. \tag{4.23}$$

For the correct class $i = GT$,

$$\begin{aligned}\frac{\partial s_{GT}}{\partial Y_{GT}} &= \frac{\partial}{\partial Y_{GT}}\frac{e^{y_{GT}}}{\sum_{i=1}^{K} e^{y_i}} = \frac{e^{y_{GT}}\sum_{i=1}^{K} e^{y_i} - e^{y_{GT}}e^{y_{GT}}}{\left(\sum_{i=1}^{K} e^{y_i}\right)^2} \\ &= \frac{e^{y_{GT}}}{\sum_{i=1}^{K} e^{y_i}}\frac{\left(\sum_{i=1}^{K} e^{y_i} - e^{y_{GT}}\right)}{\sum_{i=1}^{K} e^{y_i}} = s_{GT}(1 - s_{GT}).\end{aligned} \tag{4.24}$$

And for wrong class $i \neq GT$,

$$\frac{\partial s_{GT}}{\partial Y_i} = \frac{\partial}{\partial Y_i}\frac{e^{y_i}}{\sum_{i=1}^{K} e^{y_i}} = \frac{0 - e^{y_{GT}}e^{y_i}}{\left(\sum_{i=1}^{K} e^{y_i}\right)^2} = -\frac{e^{y_{GT}}}{\sum_{i=1}^{K} e^{y_i}}\frac{e^{y_i}}{\sum_{i=1}^{K} e^{y_i}} = -s_{GT}s_i. \tag{4.25}$$

Hence, the gradient $\nabla_Y CE$ is:

$$\nabla_Y CE = -\frac{1}{s_{GT}}\frac{\partial s_{GT}}{\partial Y} = [s_1, s_2, \dots, s_{GT} - 1, \dots, s_K] = s_i - c_i. \tag{4.26}$$

Regarding the derivatives of common activation functions, which are Sigmoid, tanh, and ReLU, the derivatives are given below, and the figure them as shown in Fig. 4.3.

$$\begin{aligned}Sigmoid'(x) &= \frac{e^{-x}}{(1 + e^{-x})^2} = \frac{1}{(1 + e^{-x})}\left(1 - \frac{1}{(1 + e^{-x})}\right) \\ &= Sigmoid(x)\big(1 - Sigmoid(x)\big).\end{aligned} \tag{4.27}$$

$$\begin{aligned}\tanh'(x) &= \frac{(e^x + e^{-x})(e^x + e^{-x})}{(e^x + e^{-x})^2} - \frac{(e^x - e^{-x})(e^x - e^{-x})}{(e^x + e^{-x})^2} \\ &= 1 - \left(\frac{e^x - e^{-x}}{e^x + e^{-x}}\right)^2 = 1 - \tanh^2(x)\,.\end{aligned} \tag{4.28}$$

$$ReLU'(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases}. \tag{4.29}$$

Once the gradient information of each neuron is obtained, the neuron can be updated using the gradient descent method and its variants, this is also called the optimizer. The vanilla version (The word "vanilla" here means ordinary, with no special or extra modified. This term is often used in the field of machine learning to refer to the original version, as described in Franklin (2005)) of the gradient descent optimizer uses all samples to calculate the loss and gradients. It introduces a hyperparameter, the learning rate lr, as the step size for each update. This process is illustrated in Eq. 4.30:

$$W^{(l)} \leftarrow W^{(l)} - lr\nabla_{W^{(l)}}. \tag{4.30}$$

This algorithm is a common method for solving unconstrained optimization problems in optimization theory. It performs well for shallow neural networks in the machine learning era. However, since it uses all samples to calculate the gradient, for a network with m neurons and N samples, each update requires at least to compute the multiplication of $N \times m$ matrices. When the amount of training data and the number of network neurons are large, the algorithm's efficiency becomes very low, and it requires significant memory to store that matrix, which makes it impractical for programming. As a practical alternative, Stochastic Gradient Descent (SGD) randomly divides the total samples into a number of subsets (called batches) and iteratively uses the samples from each batch to calculate the loss and gradients for each iteration. Once the batch gradients have been calculated, the weights of neurons are updated. Although using a small number of samples does not guarantee that the update direction for each iteration is optimal, Nemirovski et al. (2009) have shown that SGD can converge to the result of vanilla gradient descent in a probabilistic manner. This means that with a sufficient number of iterations, the loss will still converge to its minimum value. For SGD, the learning rate lr becomes a crucial hyperparameter. A learning rate that is too large can cause the algorithm to diverge, while a learning rate that is too small can result in extremely slow updates or even stagnation. To mitigate the impact of an imprecisely configured learning rate on the algorithm, two strategies have been developed to enhance its versatility.

1. **Learning Rate Decay Strategy**: This strategy involves using a larger learning rate in the early stages of training to quickly approach the minimum point with large steps. As the number of iterations increases, the learning rate is gradually reduced to ensure the algorithm converges efficiently.
2. **Improving the Update Direction**: This strategy aims to combat the oscillations caused by gradient differences across samples in different batches. It leads to the development of various SGD variants as described following.

One of the variants of SGD is SGD with momentum, the update direction of this method is a linear combination of the current gradient and the last updated gradient. For the iteration t, the formulas of updates are:

$$M^t \leftarrow lr\nabla^t_{W^{(l)}} + \alpha M^{t-1}, \tag{4.31}$$

$$W^{(l)} \leftarrow W^{(l)} - M^t, \tag{4.32}$$

where $\nabla^t_{W^{(l)}}$ is the batch gradients of the l-th layer neurons at iteration t, and α is the decay factor between 0 and 1. Combined with the gradient of the previous iteration, the update tends to keep traveling in the same direction and, therefore, can prevent oscillations. In case the iteration falls into local minima, even though the gradient at local minima is 0, the momentum M^t still has positive contributions that keep the algorithm jumping out of the minima until converges. The configuration of the hyperparameter α will also significantly affect the convergence speed of the algorithm.

For SGD with momentum, the learning rate for each neuron is consistent. This consistency can cause neurons with smaller gradients to update slowly. To solve this problem, the AdaGrad (Duchi et al., 2011) algorithm proposes adjusting the learning rate for each neuron according to its own gradient, thereby accelerating convergence. It still has a base learning rate lr, but the actual step size is the lr divided with a scaling factor from the diagonal of the outer product of a historical sum of gradient squares G. For the iteration t, the formulas of updates are:

$$G \leftarrow \sum_{\tau=1}^{t} \nabla_{W^{(l)}}^{t} {\nabla_{W^{(l)}}^{t}}^{T} \tag{4.33}$$

$$W^{(l)} \leftarrow W^{(l)} - lr\mathrm{diag}(\mathrm{G})^{-\frac{1}{2}} \nabla_{W^{(l)}}^{t}. \tag{4.34}$$

Hence, for the neuron w_i, the actual step size $\frac{lr}{\sqrt{G_i}} = \frac{lr}{\sqrt{\sum_{\tau=1}^{t}(\nabla_{w_i}^{\tau})^2}}$ is the lr multiple with the l_2 norm of the historical sum of derivatives, therefore the neuron which have big cumulative updates will be suppressed, and the neuron with small cumulative updates will have a larger actual step size for obvious updating. This adaptive method has been proven to accelerate the solving of non-convex optimization problems (Gupta et al., 2014). However, it also introduces two drawbacks: additional memory space is required to store historical gradients, and in extreme cases, the cumulative suppression may cause the actual step size to become too small, potentially terminating the algorithm prematurely before reaching the minima. In order to improve the second drawback, Root Mean Square Propagation (RMSProp) was proposed (Lecun et al., 2015). Its idea is still to penalize the base learning rate, consistent with AdaGrad, but instead of using all historical gradients, the concept of forgetting is introduced by performing a moving average on the scaling factor. For the iteration t, the formulas of updates are:

$$v^{t} \leftarrow (1-\gamma_f)\left(\nabla_{W^{(l)}}^{t}\right)^2 + \gamma_f v^{t-1}, \tag{4.35}$$

$$W^{(l)} \leftarrow W^{(l)} - \frac{lr}{\sqrt{v^t}} \nabla_{W^{(l)}}^{t}, \tag{4.36}$$

where γ_f is the forgetting factor between 0 and 1 to control the historical gradient effects, and square and square-rooting are done element wise.

The Adaptive moment estimation (Adam) optimizer integrates the advantages of the above optimizers and combines the momentum term on the RMSProp to perform adaptive dynamic smooth adjustment of the actual step size of each neuron. For the iteration t, the formulas of updates are:

$$m^{t} \leftarrow \beta_1 m^{t-1} + (1-\beta_1)\nabla_{W^{(l)}}^{t}, \tag{4.37}$$

$$v^{t} \leftarrow \beta_2 v^{t-1} + (1-\beta_2)\left(\nabla_{W^{(l)}}^{t}\right)^2, \tag{4.38}$$

$$\widehat{m} = \frac{m^t}{1-\beta_1}, \tag{4.39}$$

$$\hat{v} = \frac{v^t}{1-\beta_2}, \tag{4.40}$$

$$W^{(l)} \leftarrow W^{(l)} - lr\frac{\widehat{m}}{\sqrt{\hat{v}}+\varepsilon}, \tag{4.41}$$

where β_1 and β_2 are the forgetting factors for gradients and second moments of gradients between 0 and 1, ε is a small number to prevent division by 0, and square and square-rooting are done element wise. Compared to SGD, Adam dynamically adjusts the actual step size using the ratio of the first-order moment estimate m^t to the second-order moment estimate v^t, at the cost of twice the additional memory requirements to store both m^t and v^t. This approach makes the algorithm very stable, and typically the default parameters ($\beta_1 = 0.9, \beta_2 = 0.999, \varepsilon = 10^{-8}$) yield good results. Moreover, Adam mitigates SGD's sensitivity to the learning rate to some extent. The actual update step size is the base learning rate lr times the ratio of the first and second order of the accumulated moment estimate, which effectively makes it SNR. This adjustment allows Adam to tolerate larger improperly configured learning rate and quickly scale it to a reasonable range. The SNR represents the uncertainty between the mean of the batch gradients and the true gradient direction. For the initial iterations, larger mean gradients indicate a higher SNR, enabling the algorithm to converge quickly with a larger step size. Near the minima, a smaller mean gradient indicates that the uncertainty of the mean and the actual gradient becomes larger, prompting the algorithm to update with a smaller step size, effectively reflecting automatic annealing. This excellent property makes Adam as an optimizer work well with default parameters.

4.4. Convolutional networks and Recurrent networks

The previous section briefly introduced the general form of deep learning. Ideally, MLP with enough parameters can learn an ideal feature representation. However, in the real world, training large models with enough parameters often incurs very high costs due to hardware limitations. Therefore, finding an effective sparse representation of neurons for a specific problem, that is, determining the optimal architecture of the neural network, is crucial to the performance of the model. In this section, two well-known neural network architectures and their representative networks are introduced: convolutional neural networks (CNN) for processing data with grid-like topology and recurrent neural networks (RNN) for sequence modeling.

The motivation for CNN can be summarized as follows: for grid-like topology data, such as images, features can often be effectively extracted from local regions and their combination rather than the entire grid. This approach aligns with human intuition and is consistent with neurophysiological findings about the structure of the mammalian visual nervous system (Hubel & Wiesel, 1968). This property is very common in natural images and is particularly important in satellite remote sensing images. Take the Sentinel-1 image of Berlin Brandenburg Airport as an example, as shown in Fig. 4.6. If the task is to detect this airport, according to Fig. 4.2, each neuron in the MLP will connect all the pixels in the image, but only the tracks (red rectangle) will contribute effective information, and other pixels are almost irrelevant to this task, which will cause most of the trained weights to be almost 0, thus forcing the network to be sparsely represented. In matrix form, a neuron in MLP can be viewed as a weight matrix with the same shape as its input image or feature map, and this weight matrix should be trained to be sparse to suppress irrelevant inputs. Since the weight matrix is sparse, most of the parameters that are approximately 0 have no practical effect, so its display represents a huge waste of storage space and reduces network performance.

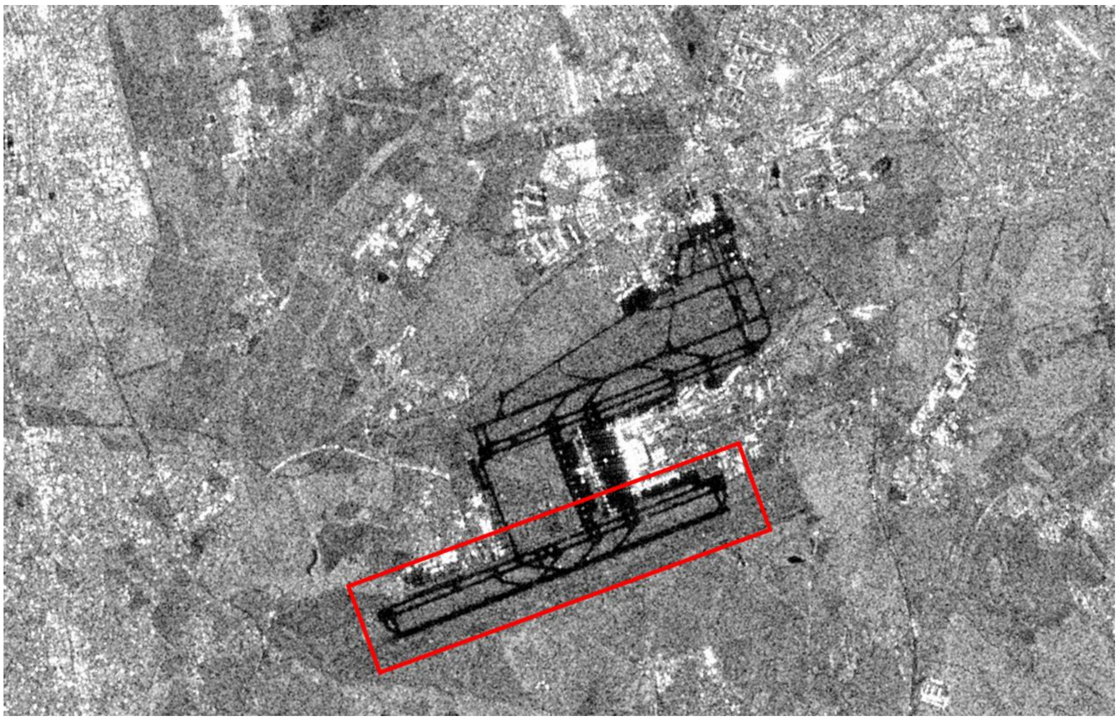


Fig. 4.6 Sentinel-1 amplitude image with VV polarization of Berlin Brandenburg Airport at 2021.12.31. The red rectangle marks the tracks.

CNN provides a way to avoid the explicit representation of sparse weight matrices by using sparse interactions. They use a series of stacked weight matrices (e.g., 3×3 or 5×5), called convolution kernels, which are multiplied with the input image or feature map in a sliding window manner and then are activated by the activation function, as shown in Fig. 4.7. Each convolution kernel can be regarded as a feature extractor, and the matrix obtained after the sliding window operation is called a feature map. This feature map represents the extraction result of the specified feature. In CNN, sparse interactions are achieved by sliding convolution kernels, which allows storing several small matrices instead of a large sparse matrix. Each trained convolution kernel can extract small but meaningful spatial features (although not necessarily easy to describe in human language). This means that CNN can significantly increase the number of feature extractors, extract more features and improve network performance even with limited memory and computing power. To extract more representative features, a downsampling operation called pooling is usually performed after the convolution and activation operations. This downsampling operation replaces the value of a certain position of the feature map with a nearby statistical feature, such as the maximum value (max pooling) or the average value (average pooling). The pooling operation is only sensitive to numerical features within the window range, which means that the learned mapping remains unchanged under small spatial translations. Using the pooling operation can reduce the size of the feature map, thereby reducing the amount of computation, and force the feature extractor to learn more spatially invariant features, improving the accuracy of feature extraction. For example, when detecting airports, the specific coordinates of the airport in the image are not as important as the relative coordinates of the runway pixels. The pooling operation encourages the feature extractor to associate consecutive runway pixels in the window with the airport, while ignoring the direction and position of the runway. Therefore, a common convolutional layer consists of three operations: convolution, nonlinear activation, and pooling. CNN consists of multiple convolutional layers of different sizes stacked together to extract spatial features at various levels. Next, a representative CNN structure U-Net will be introduced for extracting pixel-level features useful for remote sensing and geodetic tasks (Ronneberger et al., 2015).

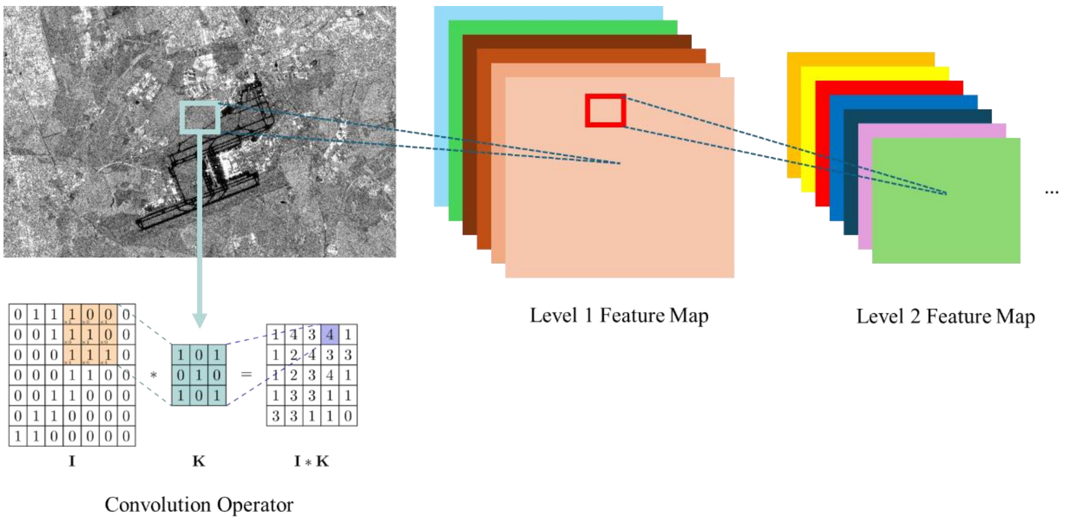


Fig. 4.7 Basic architecture of CNN, I is the input image or feature map, and K is the convolution kernel.

U-Net was originally used for the semantic segmentation of medical images, but its structure is also effective for satellite imagery used in remote sensing and geodesy. Regardless of whether the task is semantic segmentation, object classification, or mask generation, the goal is to generate a discrete probability distribution for each pixel and use this distribution to determine the category of the pixel. U-Net designs a 5-layer symmetrical encoder-decoder structure to extract spatial features at different levels. Each layer of the encoder or decoder consists of two multi-channel 3×3 convolution operations activated by ReLU, followed by a downsampling or upsampling operation. For each encoder, it first performs convolution operations to double its feature channels. Then, it downsamples using a 2×2 max pooling operation, which reduces the size of the feature map by half. Correspondingly, for each decoder, it first pads the surrounding of each pixel from the input feature map with 0 and performs a 2×2 convolution to double its size to match the size of the encoder features at the same level. This upsampling is referred to as up-convolution, deconvolution, or transposed convolution. Importantly, U-Net introduces a cross-connected structure. After upsampling, each decoder concatenates the un-downsampled features of the encoder at the same level. This step merges the shallow features extracted by the encoder with the deep features extracted by the decoder. A convolution operation is then performed to halve the feature channels. Finally, the output feature map can be converted into a discrete probability distribution by integrating its feature channels into the number of categories through a 1×1 convolution combined with the SoftMax function, thereby achieving pixel-by-pixel classification. The architecture of U-Net is shown in Fig. 4.8.

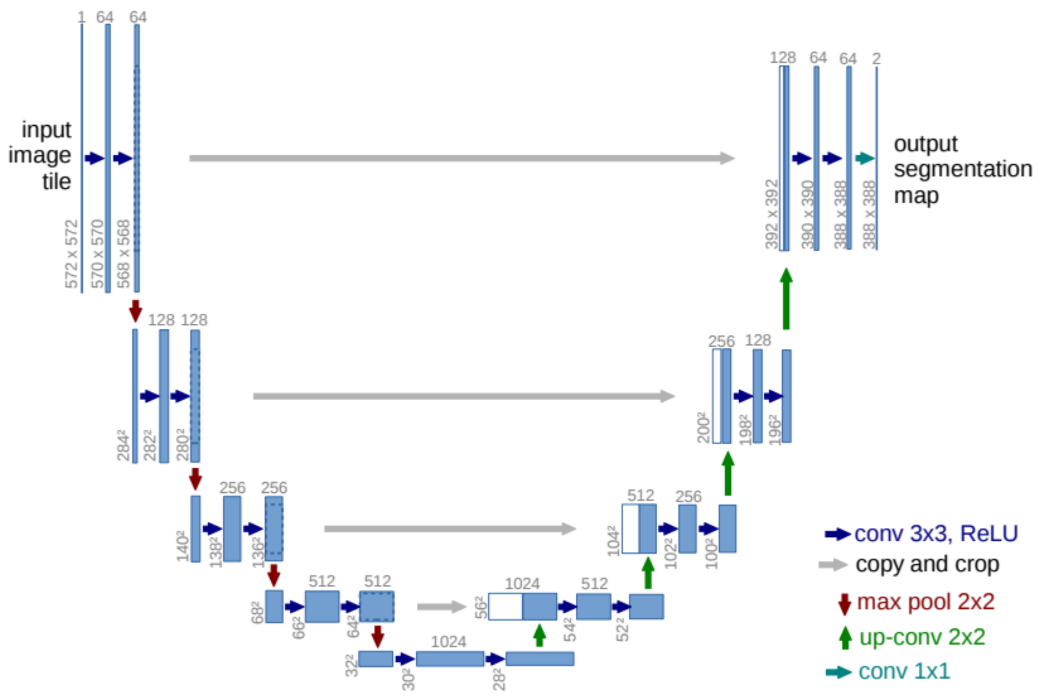


Fig. 4.8 The architecture of U-Net. Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The shape of the feature map is provided at the lower left edge of the box. White boxes represent concatenated feature maps. The arrows denote the different operations (Ronneberger et al., 2015).

In geodesy, in addition to spatial models, there is another important model known as the time series model or sequence model. Given the periodic changes of the Earth's surface, this model can learn historical patterns, thereby making predictions about current or future surface data which are needed by geodetic observations. Fortunately, just as CNN is good at automatically extracting spatial features, RNN can also automatically extract features from time series and build sequence modeling for prediction. RNN also originated in the 20th century and were independently developed from the fields of statistical mechanics (Brush, 1967) and neuroscience and cognitive psychology (Jordan, 1997). Unlike MLP and CNN, which use independent inputs, the input of RNN is a sequence of several related inputs, denoted as $X = \{x^0, x^1, x^2, \dots, x^t, \dots x^T\}$, where the superscript indicates the ordinal number of the input in the sequence. To extract features not only from the current input x^t but also to consider the impact of historical inputs, RNN expand on the concept of MLP by introducing a simple but powerful idea, which let the input to MLP includes not only the current input x^t but also the output of this MLP at the previous time step, known as hidden state h^{t-1}, as shown in Fig.4.9.

The calculation process of RNN can be seen as an extension of MLP. To accept the input x^t at time t and the historical hidden state h^{t-1} as input information together for the neuron, additional weight matrices V and U are introduced to form a linear combination a^t between x^t and h^{t-1} as input feature, followed by a hyperbolic tangent nonlinear activation to integrate the features at time t. Subsequently, MLP is used to extract high-order features from the acquired features to obtain the output, as illustrated in Eq. 4.42 to Eq.4.44:

$$a^t = Vh^{t-1} + Ux^t + b, \tag{4.42}$$

$$h^t = tanh\, a^t, \tag{4.43}$$

$$o^t = f_{DNN}(h^t; W), \tag{4.44}$$

where b is the bias vector, o^t is the output at time t, and $f_{DNN}(h^t; W)$ is a MLP with weight parameters W and input feature h^t.

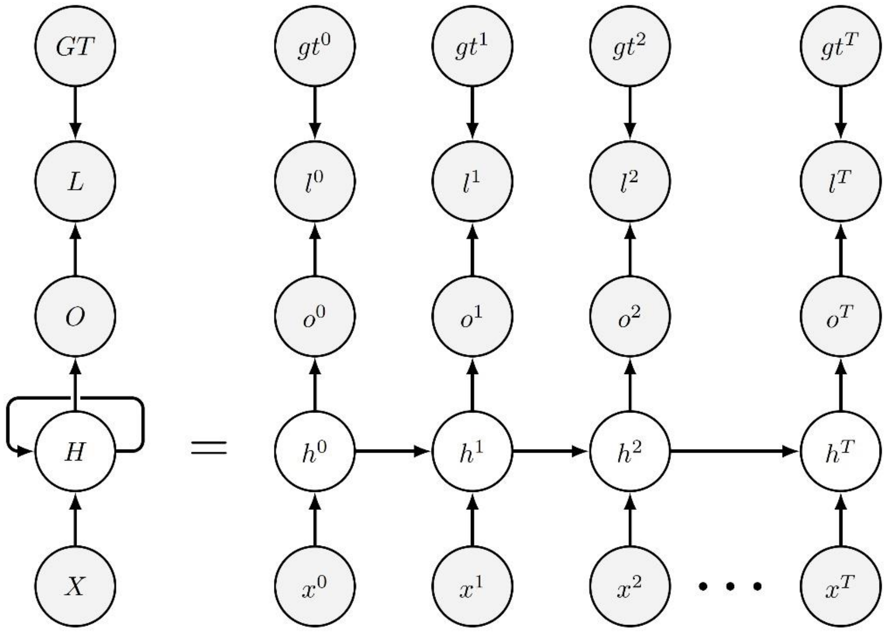


Fig. 4.9 The computational graph of RNN. $X = \{x^0, x^1, x^2, \dots, x^t, \dots, x^T\}$ is the input sequence, in this sequence $x^0, x^1, x^2, \dots, x^t$ in this figure denote the input at the corresponding time step $0,1,2, \dots t$. $H = \{h^0, h^1, h^2, \dots, h^t, \dots, h^T\}$ is the hidden states sequence, in this sequence h^0 is the initial hidden status, usually set as 0. For the following $h^1, h^2, \dots, h^t$ are the hidden status matrix for time step $1,2, \dots t$, which calculated through Eq. 4.42 and Eq. 4.43. $O = \{o^0, o^1, o^2, \dots, o^t, \dots, o^T\}$ is the output sequence of RNN at each time step, in this sequence, each o with superscript denotes the output of the network at its corresponding time step. $L = \sum_t l^t$ is the total loss and l with superscript is the loss for each time step, $GT = \{gt^1, gt^2, \dots, gt^t, \dots, gt^T\}$ is the ground truth sequence, in this sequence, each gt with superscript is the ground truth for calculating the loss at each time step.

The total loss for a sequence is the sum of loss at each time step, as shown in Eq. 4.45:

$$L = \sum_{t=1}^{T} l^t . \tag{4.45}$$

From Eq. 4.42 to Eq.4.44, it can be known that the RNN is an iterative process over time steps. Therefore, for each time step, it is necessary to independently calculate the loss with respect to the ground truth gt^t corresponding to that time step. This process is known as back-propagation through time (BPTT) (Rumelhart et al., 1986). Since the output of an RNN is generated by f_{DNN}, its loss function and the gradient update methods for all weight matrices except $\frac{\partial L}{\partial V}$ and $\frac{\partial L}{\partial U}$ can refer to Section 4.3, the derivative of MLP part $\frac{\partial L}{\partial W}$ is given below:

$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial l^t}{\partial W} = \sum_{t=1}^{T} \nabla_{\mathrm{W}}^{t} f_{DNN} . \tag{4.46}$$

where ∇_{W}^{t} is the gradient regarding as W of MLP, can be obtained by using BP described in Eq. 4.16 to Eq. 4.20. However, due to h^t depends on h^{t-1}, which is a function of all weight matrices (U, V, W), the derivatives $\frac{\partial L}{\partial V}$ and $\frac{\partial L}{\partial U}$ cannot be calculated in a straightforward manner.

For the last time step T, the gradient part regarding h^T of MLP $\frac{\partial l^T}{\partial h^T} = \nabla_h^T f_{DNN}$ can also be obtained through Eq. 4.16 to Eq. 4.20, however, for $1 \leq t < T$, the derivative becomes:

$$\frac{\partial l^t}{\partial h^t} = \frac{\partial l^{t+1}}{\partial h^{t+1}} \frac{\partial h^{t+1}}{\partial h^t} + \nabla_h^t f_{DNN} = V^{\intercal} \frac{\partial l^{t+1}}{\partial h^{t+1}} diag(1 - (h^{t+1})^2) + \nabla_h^t f_{DNN}, \tag{4.47}$$

where $diag(1 - (h^{t+1})^2)$ indicates the diagonal matrix containing the elements $1 - (h^{t+1})^2$, this is the Jacobian of the hyperbolic tangent associated with h^{t+1}. This is a very complex recursive expression. If the sequence is too long, the expanded form of Eq. 4.47 will contain $(V^{\intercal})^{T-t+1}$, which may become ill-conditioned, causing the gradient to either vanish or explode. Therefore, in this thesis, the part involving RNN typically avoids directly inputting sequences that are too long, e.g. $T \leq 24$. Once $\frac{\partial l^t}{\partial h^t}$ is obtained, the $\frac{\partial L}{\partial V}$ and $\frac{\partial L}{\partial U}$ then can be derived by:

$$\frac{\partial L}{\partial V} = \sum_{t=1}^{T} diag(1 - (h^t)^2) \frac{\partial l^t}{\partial h^t} (h^{t-1})^{\intercal}, \tag{4.48}$$

$$\frac{\partial L}{\partial U} = \sum_{t=1}^{T} diag(1 - (h^t)^2) \frac{\partial l^t}{\partial h^t} (x^t)^{\intercal}. \tag{4.49}$$

Since the computation is iteratively, intermediate variables such as $\frac{\partial l^t}{\partial h^t}, h^t, \nabla_h^t f_{DNN}$ are stored to avoid repeated calculations.

Eq. 4.47 shows that the derivative $\frac{\partial l^t}{\partial h^t}$ depends not only on the gradient part originating from the MLP at time t $\nabla_h^t f_{DNN}$, but also on the product integral of the derivatives $\prod_t^{T-1} \frac{\partial h^{t+1}}{\partial h^t}$. Expanded this term, there is:

$$\prod_{t}^{T-1} \frac{\partial h^{t+1}}{\partial h^t} = \prod_{t}^{T-1} V^{\intercal} diag(1 - (h^{t+1})^2). \tag{4.50}$$

Since h^{t+1} is activated by hyperbolic tangent, it must be smaller than 1, then the elements of $diag(1 - (h^{t+1})^2)$ must be greater than 0 but smaller than 1. As the length of the sequence between the current time step t and the final time step T increases, its gradient will be multiplied by a matrix with a value less than 1 but greater than 0. When there are many consecutive multiplications, this causes the gradient to approach 0 for the time step far from the final time step, failing to transmit gradient information effectively across by time. This not only leads to the vanishing gradient problem but also causes RNN to focus on learning time-domain features from time steps close to the final step, failing to capture long-term dependency features (Pascanu et al., 2013). To address this issue, the Long Short-Term Memory Network (LSTM) was proposed (Hochreiter & Schmidhuber, 1997). LSTM uses the Sigmoid function (Eq.4.3) to construct three gate structures, which expands the linear combination part (Eq.4.42 and Eq.4.43) in RNN to control the propagation of information and its gradients. Specifically, LSTM expands the hidden state propagate as follows:

$$f_g^t = \sigma_{sig}\left(W_f x^t + U_f h^{t-1} + b_f\right) \tag{4.51}$$

$$i_g^t = \sigma_{sig}(W_i x^t + U_i h^{t-1} + b_i) \tag{4.52}$$

$$o_g^t = \sigma_{sig}(W_o x^t + U_o h^{t-1} + b_o) \tag{4.53}$$

$$\tilde{c}^t = tanh(W_c x^t + U_c h^{t-1} + b_c) \tag{4.54}$$

$$c^t = f_g^t \odot c^{t-1} + i_g^t \odot \tilde{c}^t \tag{4.55}$$

$$h^t = o_g^t \odot tanh(c^t) \tag{4.56}$$

where $f_g^t, i_g^t, o_g^t, \tilde{c}^t, c^t$ are the forget gate's activation matrix, input gate's activation matrix, output gate's activation matrix, candidate cell memory matrix, and cell state matrix, respectively at time step t. $W, U,\ b$ with its superscript represent the weight matrices and biases which need to be learned associated with the respective gates, σ_{sig} is the Sigmoid function described in Eq.4.3, $\odot$ denotes the Hadamard product. The comparison between an RNN unit and an LSTM unit is illustrated in Fig. 4.10.

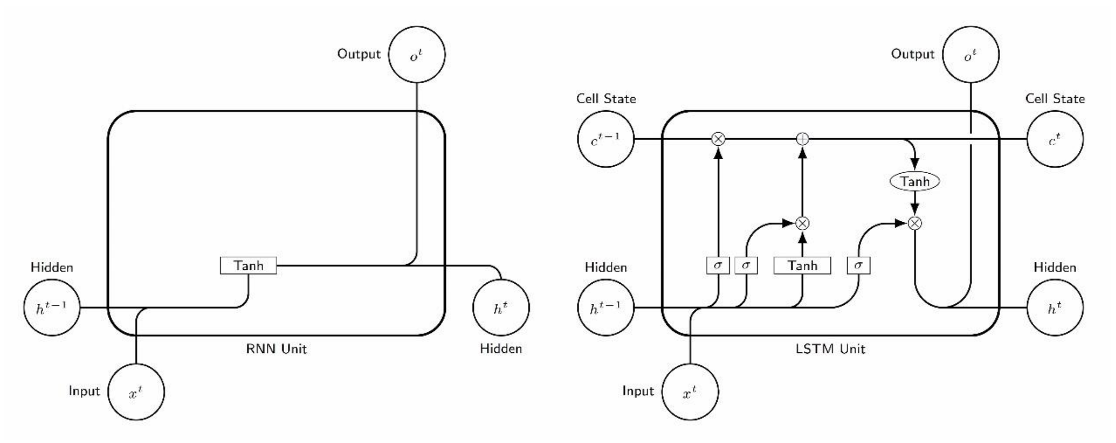


Fig. 4.10 The RNN Unit (Left) and the LSTM Unit (Right).

For RNN units, the hidden state h^t is directly controlled by the previous hidden state h^{t-1}, which is the main reason for the vanishing gradient problem and the inability to capture long-term dependencies. In contrast, for LSTM units, the hidden state h^t is determined by the output gate o_g^t, which contains information from the previous hidden state h^{t-1} and the cell state c^t. The cell state c^t is determined by the previous cell state c^{t-1}, the forget gate, the input gate, and the candidate cell memory together which use the previous hidden state h^{t-1} together. This means that, in addition to the hidden state propagation path, LSTM has an additional cell state propagation path to better retain and propagate information over long sequences, effectively addressing the vanishing gradient problem and enabling the capture of long-term dependencies. From Eq. 4.50, it can be known that the recursive gradient is the culprit for the above problem. For LSTM, the recursive gradient happened in:

$$\frac{\partial l^t}{\partial W_f} = \sum_{i=1}^{t} \frac{\partial l^t}{\partial o^t} \frac{\partial o^t}{\partial h^t} \frac{\partial h^t}{\partial c^t} \frac{\partial c^t}{\partial c^i} \frac{\partial c^i}{\partial W_f}, \tag{4.57}$$

and the recursive part become:

$$\frac{\partial c^t}{\partial c^i} = \frac{\partial c^t}{\partial c^{t-1}} \frac{\partial c^{t-1}}{\partial c^{t-2}} \cdots \frac{\partial c^{i+1}}{\partial c^i}. \tag{4.58}$$

For the single one of those terms by taking the derivative of c^{t+1} with respect to c^t, there is:

$$\begin{aligned}\frac{\partial c^t}{\partial c^{t-1}} &= \frac{\partial c^t}{\partial f_g^t} \frac{\partial f_g^t}{\partial h^{t-1}} \frac{\partial h^{t-1}}{\partial c^{t-1}} + \frac{\partial c^t}{\partial i_g^t} \frac{\partial i_g^t}{\partial h^{t-1}} \frac{\partial h^{t-1}}{\partial c^{t-1}} + \frac{\partial c^t}{\partial \tilde{c}^t} \frac{\partial \tilde{c}^t}{\partial h^{t-1}} \frac{\partial h^{t-1}}{\partial c^{t-1}} + f_g^t \\ &= c^{t-1} \sigma'_{sig}\left(W_f x^t + U_f h^{t-1} + b_f\right) U_f o_g^{t-1} \odot tanh'(c^{t-1}) \\ &+ \tilde{c}^t \sigma'_{sig}(W_i x^t + U_i h^{t-1} + b_i) U_i \odot o_g^{t-1} tanh'(c^{t-1}) \\ &+ i_g^t tanh'(W_c x^t + U_c h^{t-1} + b_c) U_c \odot o_g^{t-1} tanh'(c^{t-1}) \\ &+ f_g^t. \end{aligned} \tag{4.59}$$

The last term in Eq. 4.59 represents the output of the forget gate. This means that the network learns to adjust the value of the forget gate to decide when to forget the gradient and when to retain it, so that $\frac{\partial c^t}{\partial c^{t-1}}$ is not always less than 1. By doing so, it avoids gradient vanishing and preserves the gradient of long-term information, effectively maintaining the propagation of important information across long sequences.

5. Deep learning methods for tropospheric effects in space geodesy

In microwave-based space geodesy, including GNSS and InSAR, tropospheric delay is a significant source of error that affects the measurement accuracy. Direct observation of tropospheric effects in space geodetic techniques is extremely challenging, so tropospheric "modeling" must be used to simulate and remove its effects. Such modeling usually relies on meteorological data from in situ observations or NWM. However, even with state-of-the-art NWM (e.g., ERA5), the spatial resolution it provides remains at the sub-degree level (about 25 km) with centimeter-level retrieval errors, which are non-negligible for millimeter-level deformation monitoring tasks. Moreover, NWM products are usually not available in real time, which further complicates real-time water vapor sensing via GNSS in the absence of in situ observation capabilities. Therefore, a high-precision tropospheric model is urgently needed to meet the requirements of different geodetic applications.

To address the above questions, this chapter proposes two deep learning-based tropospheric modeling approaches. One approach, called GM-LSTM, aims to integrate historical NWM ray-traced ZTD and ZTD observations from GNSS, thereby providing high-precision estimates of ZTD and ZWD at any location in space. The other approach achieves high-precision water vapor retrieval in the absence of real-time NWM meteorological data by using a combined DNN and LSTM model.

The primary contents of this chapter are based on the following peer-reviewed papers, which were published or are under review during the author's doctoral studies:

Wang, D., Yuan, P., Kutterer, H. (2024). *Real-Time GNSS Integrated Water Vapor Sensing Based on Time Series Correction Deep Learning Models.* In: International Association of Geodesy Symposia. Springer, Berlin, Heidelberg.

Wang, D., Wang, L., Kutterer, H. (2025). *An advanced tropospheric delay model based on Gaussian Mixed Long Short-Term Memory Network.* IEEE Transactions on Geoscience and Remote Sensing.

5.1. Spatial Inference of Tropospheric Delay Based on Gaussian Mixed Long Short-Term Memory Network

5.1.1. Problem statement and current state

As discussed in Chapter 2, microwave signals bend and experience delays when propagating through the neutral atmosphere, which can introduce errors into space geodetic measurements. Therefore, it is crucial to develop a modeling approach that can accurately estimate the delay caused by the neutral atmosphere at any location within the study area.

To estimate this delay, a widely adopted approach is to decompose the ZTD into ZHD and ZWD. These components are then modeled separately by ray tracing using meteorological data from the NWM, as described in Section 2.3. However, this approach has inherent limitations in terms of both accuracy and spatial resolution. The centimeter-level RMSE and spatial resolution of approximately 25 km (ERA5) make it challenging to achieve the millimeter-level accuracy required for geodetic applications. To address the spatial resolution limitations of the NWM and improve the accuracy of ZTD estimates, relevant research has focused on spatial interpolation techniques, which include power-law vertical adjustment (Eqs. 2.39, 2.40, 2.51, and 2.52) and horizontal interpolation such as IDW and bilinear interpolation. However, these interpolation methods are based on a key assumption: the vertically scaled meteorological parameters (temperature, pressure, and water vapor pressure) are homogeneous within a 25 × 25 km^2 grid at each pressure level. In practical situations, this assumption is likely not met, as the resolution of 0.25° is not sufficient to adequately represent the impact of topography on near-surface atmospheric dynamics. Moreover, the turbulent and heterogeneous behavior of near-surface water vapor exacerbates this problem, making the assumption of spatial uniformity unrealistic under practical conditions. Another factor to consider is that NWM meteorological data are not direct measurements, but the output of data assimilation, combining atmospheric observations with physical models. This means that the gridded NWM data contain uncertainties themselves. In addition, the ray tracing method itself may also introduce errors, as it assumes that the atmosphere is spherically symmetric in the local area, and the determination of constants such as K1, K2, and K3 may also be subject to laboratory measurement errors. Therefore, as pointed out by Ding et al. (2023), the combined errors of interpolation and the NWM ray tracing model often lead to centimeter-level errors in ZTD estimates.

Considering that GNSS can provide reliable station-by-station ZTD products, many researchers have explored using GNSS-retrieved ZTD products to improve the accuracy of NWM ray tracing. Such enhancements mainly focus on improving interpolation techniques. For example, Jarlemark & Emardson (1998) evaluated the effects of gradient models, turbulence models, and temporal linear regression models on the interpolation of ZWD. It was concluded that the turbulence model performed at least 10% better than the other models. Janssen et al. (2004) demonstrated that the IDW and ordinary kriging interpolation methods produce better results than spline interpolation. Meanwhile, Onn & Zebker (2006) and Xu et al. (2011) proposed using the frozen flow assumption to simulate water vapor, combined with a simple variable local mean kriging estimator, which improved the interpolation accuracy by 29%. Löfgren et al. (2010) virtualized the GNSS stations to NWM grid points and used IDW and Gaussian interpolation to improve ray tracing. However, this method does not consider the impact of geospatial weights on interpolation. Therefore, it has been proven to be ineffective when the GNSS station density is sparse. Yu et al. (2018) developed a weighted integration scheme to estimate the turbulent and stratified components from the ZTD retrieved from GNSS using the ITD model (Eq. 2.58) and performed IDW interpolation on the turbulent component only. Yu et al. (2018) reported that this scheme achieved a ZTD estimation error as low as 1 cm and was made available to the public as a GACOS product. Although this method provides a geographically weighted integration strategy, it cannot use advanced mapping functions such as VMF3 to convert ZTD into slant delays because it cannot decompose ZTD into ZHD and ZWD. In addition, this strategy relies on IDW interpolation to calculate the turbulence component, which also assumes that turbulence is spatially uniform. This assumption may not hold true in weather events such as rainfall, resulting in poor performance under bad weather conditions.

Due to the complexity of water vapor activity and the influence of turbulence, accurate modeling of ZWD at any location remains an open question in the geodesy and remote sensing community. Fortunately, as a data-driven method, deep learning provides the possibility to address this issue. Osah et al. (2021) employed a deep neural network using TensorFlow and Keras to predict daily IGS final ZTD values for four selected IGS stations in West Africa, with VMF3-ZTD as input. Yang et al. (2021) applied ANN with 2-layers and 5 neurons to correct

GPT3-ZTD, demonstrating that their model outperformed the Saastamoinen and GPT3 models using data from a satellite positioning reference network in Hong Kong. In another related approach, Yang et al. (2021) further proposed a 2-layer, 4- neurons neural network to correct ZTD estimates from the Hopfield and Saastamoinen models, validating its effectiveness at 67 GNSS stations in China and showing that it significantly improved the accuracy of these traditional models. Additionally, Zhang et al. (2024) applied a 7-layer MLP to perform GNSS-derived water vapor tomography, estimating layered precipitable water in a task closely related to ZWD. These studies collectively illustrate the applicability of deep learning for ZTD estimation. However, the neural network architectures used in these efforts are relatively simple, focusing solely on regressing target values. This simplicity restricts the networks' ability to capture the complex spatiotemporal characteristics of ZTD, limiting the potential of deep learning for accurate and robust ZTD estimation.

5.1.2. Motivation

Looking back at the tropospheric delay concept itself, tropospheric delay refers to the additional path length that microwaves experience as they pass through the neutral atmosphere due to refraction and deceleration. The observation equation Eq. 2.17 strictly describes this phenomenon. In order to accurately determine the tropospheric delay, the ionospheric delay in Eq. 2.17 must be taken into account and eliminated. To achieve this, space geodetic techniques that can obtain "ionosphere-free" observations are required. These observations use more than two operating frequencies to effectively eliminate the ionospheric delay, as shown in Eq. 2.21.

Although other techniques, such as DORIS (Doppler Orbiting and Radio positioning Integrated by Satellite) and VLBI (Very Long Baseline Interferometry) also have multiple operating frequencies and can derive observation equations similar to Eq. 2.21, their available observation infrastructure (satellites, ground stations and radio telescopes) is much less than GNSS. Considering the spatial heterogeneity of the troposphere, the temperature, pressure and water vapor content vary greatly at different locations. Although observations from DORIS or VLBI are effective in retrieving the tropospheric conditions above their sites, their sparse spatial distribution may not be sufficient for spatial tropospheric modeling.

In contrast, GNSS technology has a significant advantage due to the large number of satellites in orbit. This ensures that the number of valid observations is more than 10 most of the time in most regions of the world. This allows adjustments to be made during the GNSS solution process to improve the accuracy of tropospheric delay calculations. In addition, GNSS has very complete orbit and clock products (these products can be obtained free of charge from the (IGS web server) to support GNSS-PPP for tropospheric delay solutions. In most countries or regions, there are public or private GNSS CORS networks, such as the Satellitenpositionierungsdienst der Deutschen Landesvermessung (SAPOS ®) or the EUREF Permanent GNSS Network. These networks consist of permanent GNSS stations with an average distance of tens to hundreds of kilometers in the region (depending on the operating company and the specific region) to provide GNSS positioning service references. These permanent GNSS stations are well-constructed and equipped with high-precision multi-frequency receivers and antennas to ensure the accuracy of observations. These observations can be used to obtain reliable ZTD, ZHD, and ZWD over these stations using the GNSS-PPP technique. In addition, the Nevada Geodetic Laboratory (NGL) provides the RENIX observation files and their corresponding ZTD products solved by GISPY-OASIS-II software from global public GNSS CORS networks (Blewitt et al., 2018). This well-developed product community and data accessibility ensure that there are usually several station-wise ZTD observations in the region that can be considered as ground truth.

Although using GNSS-retrieved ZTD to enhance NWM ray-tracing estimates is a promising approach, a key limitation is that the spatial density of reliable GNSS observations is still too low for spatial interpolation methods. In most cases, the distance between GNSS permanent stations is greater than 25 km. This means that when performing IDW interpolation, there may be at most only one GNSS permanent reference station in four adjacent grid points of NWM. In this case, the contribution of the GNSS permanent reference station to the IDW interpolation result may be much smaller than that of the four NWM grid points, unless the study area is close enough to the GNSS permanent reference station. Fig. 5.1 analyzes the impact of weighting on the IDW interpolation results when only one GNSS permanent reference station is present within four adjacent grids. Assume that the study area is located equidistantly from four NWM grid points, each spaced 25 km apart, placing the study area approximately 17.68 km from any of these grid points. In this scenario, if the GNSS permanent reference station is situated on a circle with a radius of 17.68 km from the study area, its contribution to the IDW interpolation is equivalent to one of the four adjacent NWM grid points. If the GNSS station is located within this circle, its contribution exceeds that of an individual NWM grid point. Since IDW interpolation involves a weighted sum of all sample points, this implies that the GNSS station will make a dominant contribution (greater than 50%) to the final interpolation result only if it is within 17.68/4=4.42 km of the study area. In the real scene, it is very challenging to ensure that a GNSS permanent reference station is within this 4.42 km range. In some regions, such as Northern Europe, the distance between GNSS permanent reference stations may range from 50 to 100 km. Consequently, without implementing additional weighting strategies, the inclusion of GNSS permanent reference stations provides only limited improvement to the interpolation results.

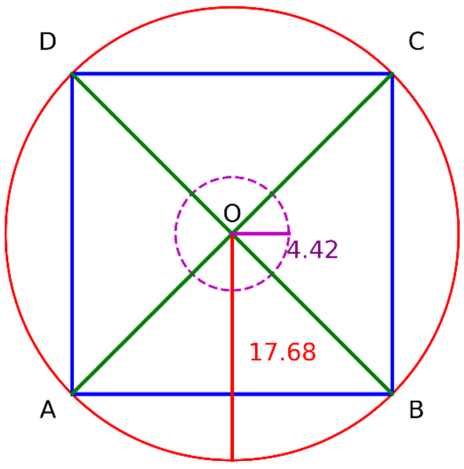


Fig. 5.1 Example of the IDW interpolation influence range: Point O represents the study area, while points A, B, C, and D are NWM grid points with a spacing of 25 km, and each located approximately 17.68 km from the study area O, providing ZTD estimates via NWM ray tracing. The red circle, with a radius of 17.68 km, delineates the area within which the weight of the GNSS-observed ZTD in IDW interpolation is equivalent to that of any ZTD from NWM grid point. The purple circle, with a radius of 4.42 km, represents the area where the GNSS observed ZTD has a dominant influence in IDW interpolation, meaning its weight exceeds 50%.

Although some studies, e.g. Yu et al. (2017) attempted to enhance the weighting strategy of IDW, the irregular and generally lower spatial density of available GNSS stations compared to NWM grid points presents a significant challenge. As a result, directly integrating the ZTD from GNSS observations into the NWM grid often leads to limited improvements. To address this issue, it is essential to find a method that does not directly integrate the GNSS observed ZTD but instead leverages its information to enhance the accuracy of the NWM ray tracing method. Considering that NWMs incorporate atmospheric physical laws and multiple observations, they typically exhibit spatial stability in the absence of extreme meteorological anomalies. This suggests that if a mapping can be established to relate the NWM ray tracing ZTD to the GNSS observed ZTD, this mapping should be applicable to any location near the observation area. In other words, by using the ZTD from GNSS observations as supervisory data, the error pattern of the NWM ray tracing method can be learned and subsequently applied to correct NWM ray tracing ZTD at any location within the effective range without the need for integration interpolation. Given that the error pattern may vary across different regions and seasons, this mapping could be highly complex and dynamic subject to temporal changes. Fortunately, with the availability of extensive public GNSS data, deep learning can be employed to directly learn the error patterns of local areas without the need for explicit mathematical formulations. This constitutes motivation for this study.

The purpose of this study is to employ deep learning to learn the correction pattern of NWM ray tracing from GNSS-observed ZTD. By applying this learned pattern, the NWM ray tracing ZTD obtained through grid interpolation can be directly corrected, thereby avoiding the spatial sparsity issues that arise by integrating GNSS-observed ZTD in spatial interpolation. Since NGL products only provide ZTD products, referred to as ZTD_{GNSS}, it is not possible to accurately separate the hydrostatic and wet components, as described in section 2.3. Considering that the VMF (Boehm et al., 2006) and the IGS solving strategy (Dach & Bockmann, 2024) typically use the Saastamoinen hydrostatic model (Eq. 2.50) as the prior for the hydrostatic delay, this study also adopts this approach. Although the hydrostatic delay may include some wet components, the absence of suitable measurement techniques to distinguish between these components necessitates accepting this prior hydrostatic delay and focusing on correcting the remaining part, which is the ZWD.

To take advantage of the VMF mapping function in converting ZTD to slant delay, this study utilizes the ERA5 pressure level product (one of the most advanced NWM) and the VMF3 gridded product as data sources. The ERA5 pressure level product[3] (Hersbach et al., 2023) provides meteorological data across 37 pressure levels from the surface to the tropopause at a resolution of 0.25° every hour. This data can be used to calculate the ZWD at any location using Eqs. 2.32-2.42, referred to as ZWD_{ERA5}. The VMF3 grid product[4] (Landskron & Böhm, 2018), on the other hand, offers surface ZHD, ZWD, and their corresponding mapping coefficients every 6 hours at a spatial resolution of 1° grid. To ensure compatibility with the VMF3 mapping coefficients, this study employs the same spatiotemporal interpolation strategy as the VMF3 grid product, as described in Eqs. 2.43-2.52, referred to as ZHD_{VMF3} and ZWD_{VMF3}. Given that the 6-hour temporal resolution of VMF3 is coarser than the 1-hour resolution of ERA5, all VMF3-derived products are subjected to linear temporal interpolation between two surrounding epochs to match the corresponding time of ERA5. Since ZHD_{VMF3} is fundamentally based on the Saastamoinen hydrostatic model—the same model used for the a priori delay in GNSS solutions—it is assumed that the residual between ZTD_{GNSS} and ZHD_{VMF3} represents the wet delay retrieved by GNSS ZWD_{GNSS}, as described below:

$$ZWD_{GNSS} = ZTD_{GNSS} - ZHD_{VMF3}. \tag{5.1}$$

The objective of the model is to find a mapping f that transforms the input $ZWD_{ERA5}$and

[3]Data can be accessed at: https://cds.climate.copernicus.eu/cdsapp#!/dataset/reanalysis-era5-pressure-levels?tab=overview

[4]Data can be accessed at: https://vmf.geo.tuwien.ac.at/

ZWD_{VMF3} into ZWD_{GNSS}, as described below:

$$f(ZWD_{ERA5}, ZWD_{VMF3}) \rightarrow ZWD_{GNSS}. \tag{5.2}$$

This mapping f can be obtained by training a deep neural network that is supervised by ZWD_{GNSS} derived from the GNSS permanent reference station network products. Admittedly, using MLP as the network structure to directly perform numerical regression with the RMSE loss function is a straightforward approach to obtaining this mapping f. However, there are two problems with this method:

1. **Temporal Correlation:** The residual sequence between ZWD_{GNSS} and ZWD_{ERA5} or ZWD_{VMF3} does not follow Gaussian white noise, indicating the presence of temporal correlations, as shown in Fig. 5.2. MLP cannot capture temporal correlations within the sequence and therefore cannot exploit these temporal correlations to improve the accuracy of the model.

2. **Risk of Overfitting and Uncertain Output:** The mapping f trained by using data from a single GNSS permanent reference station risks overfitting, which may limit its generalization capability. This means that f supervised by ZWD_{GNSS} data from the nearest GNSS reference station may be valid only at that station location and not applicable to the wider study area. To ensure generalization, it is necessary to include multiple GNSS permanent reference stations around the study area in the training process. Unfortunately, due to the uneven distribution of GNSS stations, some stations may be more than 100 km away from the study area and have large vertical differences. In this case, the atmospheric conditions above these different GNSS stations may be different, resulting in different correction patterns. This heterogeneity may lead to different outputs for the same input, and the RMSE loss function used in numerical regression cannot handle this situation well.

To address the above issues, this study proposes a deep learning method, namely GM-LSTM, which estimates the ZWD probability density function by extracting temporal correlated features from ZWD_{ERA5} and ZWD_{VMF3}time series. By employing this probabilistic model, GM-LSTM not only provides more accurate ZWD estimates for any location within the study area but also offers uncertainty estimates related to water vapor heterogeneity. Furthermore, since GM-LSTM incorporates corrections based on VMF3 tropospheric products, it can map the ZHD and ZWD to the slant direction respectively using the coefficients provided by VMF3. This makes GM-LSTM the state-of-the-art model for ZTD estimation and tropospheric correction.

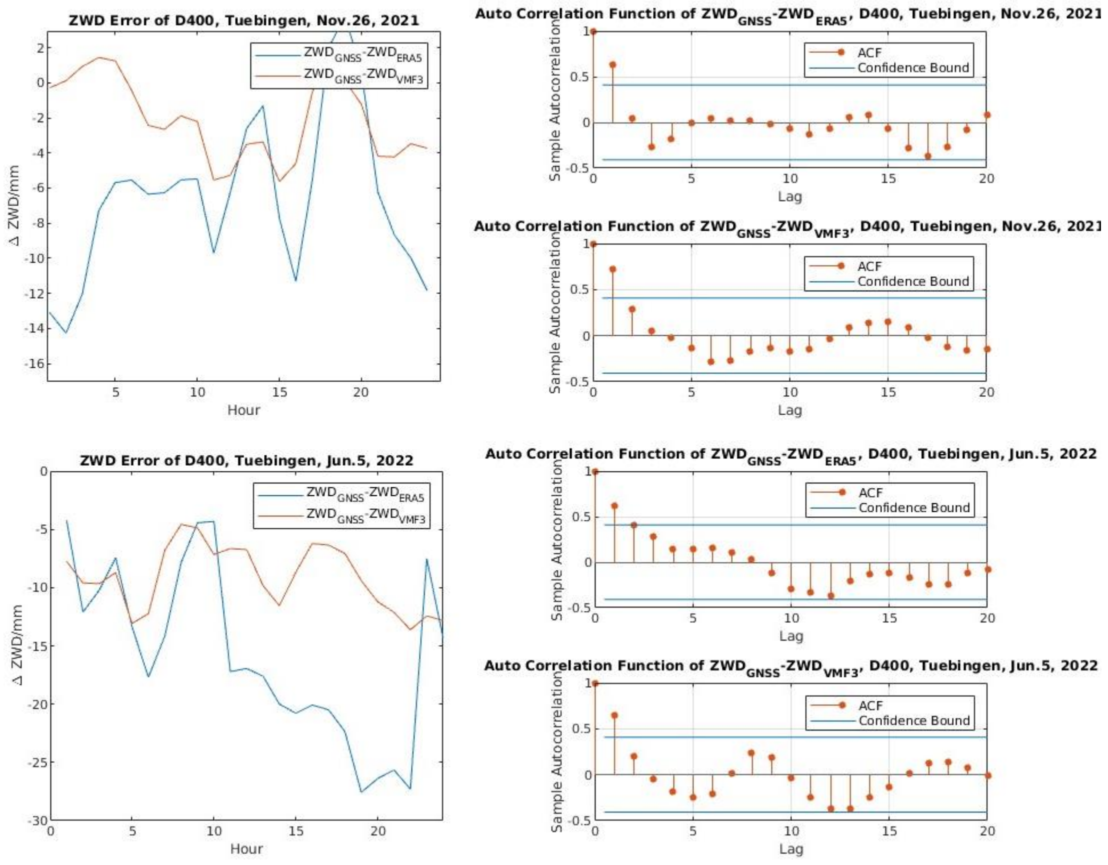
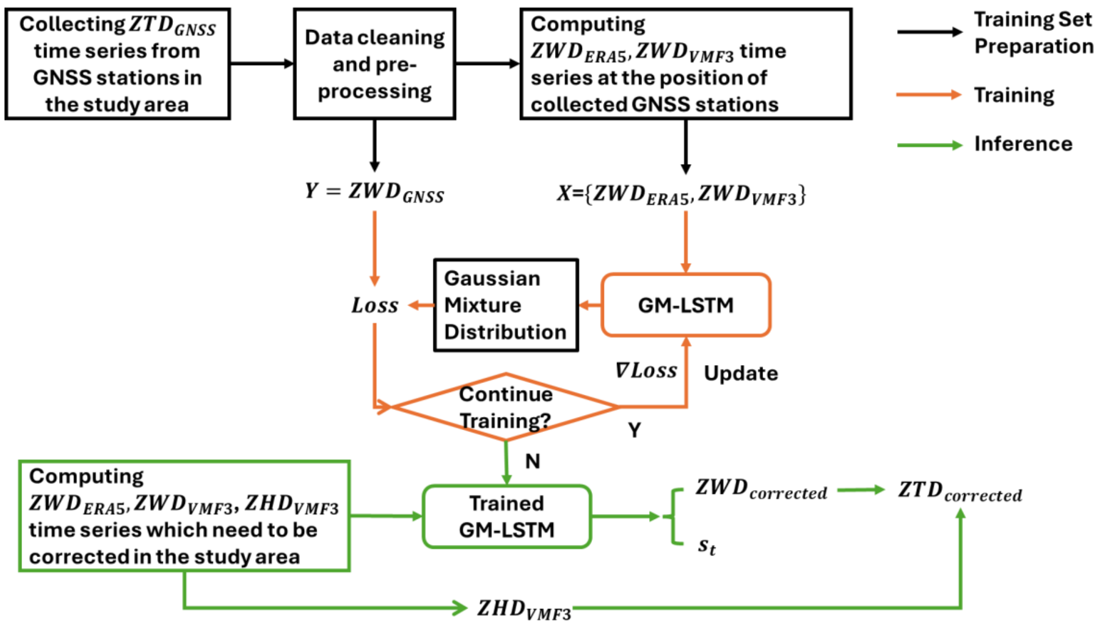


Fig. 5.2 ZWD_{ERA5} and ZWD_{VMF3} errors at GNSS station D400 in Tübingen, on Nov.26 2021, and Jun. 5, 2022. From the Auto Correlation Function (ACF), there are lags significantly above 1 no matter for winter or summer, ERA5 or VMF3, which means that at least first-order autocorrelation exists.

5.1.3. Methodology

Despite the tropospheric models, such as VMF3 being considered to provide reliable ZHD_{VMF3},a significant discrepancy remains between the modeled ZTD from VMF3 or ERA5 and the reference. This discrepancy is primarily due to the inherent challenges associated with ZWD modeling. To address this discrepancy, this research put forward employing a deep learning approach based on GM-LSTM to individually learn the mapping $\{ZWD_{ERA5}, ZWD_{VMF3}\} \rightarrow pdf(ZWD_{GNSS})$. This approach operates under the assumption that this mapping pattern is shared across a local spatial range. Under this assumption, the GM-LSTM can be trained through the supervised data ZWD_{GNSS} collected from the surrounding GNSS stations and the corresponding ZWD_{ERA5} and ZWD_{VMF3} at the same position. After training, the learned pattern can be used for the entire study area. The workflow of this approach is summarized in Fig. 5.3, and details will be in the following section.



Collecting ZTD_{GNSS} time series from GNSS stations in the study area
Data cleaning and pre-processing
Computing ZWD_{ERA5}, ZWD_{VMF3} time series at the position of collected GNSS stations
Training Set Preparation
Training
Inference
$Y = ZWD_{GNSS}$
$X=\{ZWD_{ERA5}, ZWD_{VMF3}\}$
Gaussian Mixture Distribution
Loss
GM-LSTM
$\nabla Loss$
Update
Continue Training?
Y
N
Computing ZWD_{ERA5}, ZWD_{VMF3}, ZHD_{VMF3} time series which need to be corrected in the study area
Trained GM-LSTM
$ZWD_{corrected}$
$ZTD_{corrected}$
s_t
ZHD_{VMF3}

Fig. 5.3 The workflow of ZTD modeled by GM-LSTM. s_t denotes the standard deviation of trained Gaussian Mixture Distribution, which reflects the uncertainty at time t.

5.1.3.1 Training Set Preparation

To train the model and obtain the mapping pattern of the study area, supervised data from the GNSS permanent reference stations around the study area is required. Ideally, GNSS stations closer to the study area provide more reliable supervision information. However, the density of GNSS stations cannot always be guaranteed. Fortunately, the experiments indicate that GNSS stations located within a radius of 200 km around the study area suffice to provide effective supervision information for learning. This range is typically covered by NGL products[5].

After collecting the ZTD_{GNSS} time series from GNSS stations around the study area, the following cleaning was performed to ensure the quality of the training set: Only hourly data with a daily acquisition completeness higher than 75% were collected. For occasional missing GNSS observations (daily completeness between >75% and <100%), spline interpolation was employed to ensure that the data covered the entire 24-hour period.

When cleaning is finished, use Eq. 5.1 to prepare ZWD_{GNSS} series and divide it by day. Then stack the whole collection into a $24 \times N$ matrix, where N represents the total number of valid days with all collected GNSS stations. This stack serves as the ground truth set Y for training the network. Next, using Eqs. 2.32–2.52 along with the ERA5 and VMF3 grid products, prepare ZWD_{ERA5} and ZWD_{VMF3} corresponding to the time and location of Y as the network input training set X, which form a 2-24-by-N tensor.

5.1.3.2 Training

When the training set is prepared, a GM-LSTM network can be trained as follows:

1) Initialize the GM-LSTM network

The GM-LSTM network aims to construct a Gaussian Mixture model to effectively describe

[5] Data can be accessed at: http://geodesy.unr.edu/NGLStationPages/gpsnetmap/GPSNetMap.html

the uncertainty introduced by turbulence or weather activities. By using the temporal memory and sequence modeling capabilities of the LSTM network, GM-LSTM can capture the complex dynamic characteristics of ZWD changing over time. GM-LSTM comprises three stacks of Bidirectional LSTM (Bi-LSTM) layer (Graves & Schmidhuber, 2005) that serve as its feature extractor. Each layer of Bi-LSTM passes its hidden state to the next time step. The features extracted by the three stacks of Bi-LSTM are then decoded by three independent MLP into the parameters required to describe the Gaussian mixture model: the mixing coefficients α, mean μ, and variance σ^2 of each Gaussian component. Before training, the GM-LSTM needs to be initialized according to its architecture, as illustrated in Fig. 5.4.

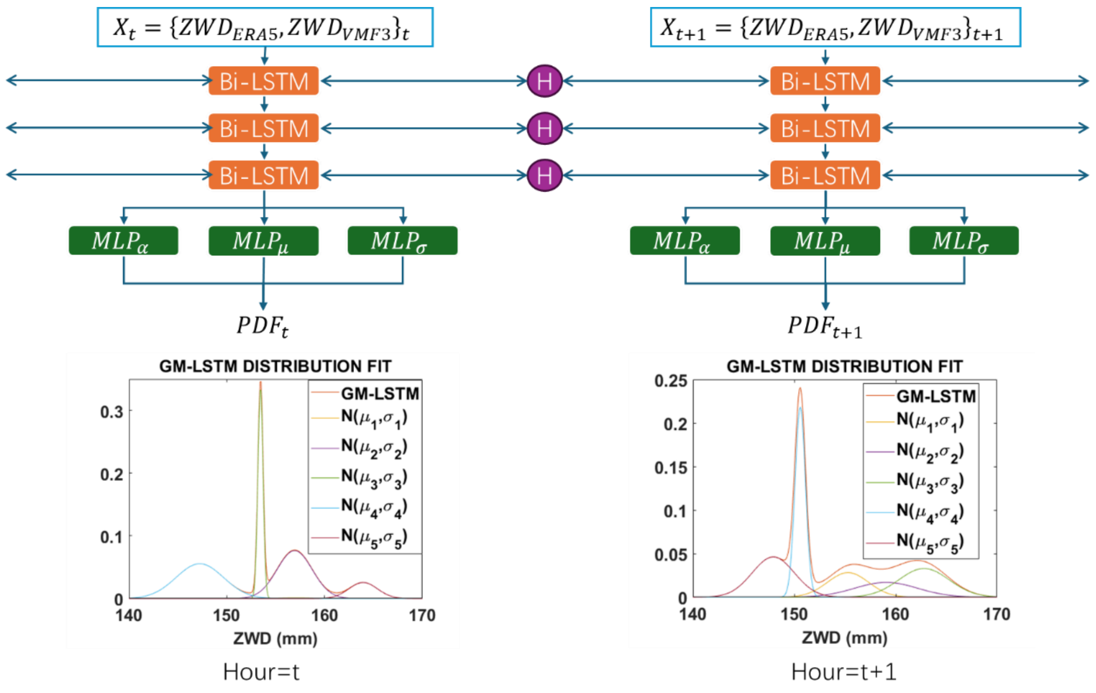


Fig. 5.4 Architecture of GM-LSTM. H is a hidden state that contains historical information, and PDF_t denotes the probability density function of the Gaussian Mixture Distribution inferred by GM-LSTM at time.

Bi-LSTM is an extension of the LSTM unit described in Section 4.4. Each Bi-LSTM layer contains two independent LSTM units: one extracts forward time series features from 00:00 to the current time, while the other extracts backward time series features from 23:00 to the current time. In the Bi-LSTM layer, the forward LSTM unit and backward LSTM unit maintain their own hidden state $h_t^{\rightarrow}$ and $h_t^{\leftarrow}$, and the output of the Bi-LSTM unit H_t is the concatenation of these two hidden states, as illustrated in Fig. 5.5. The reason for using Bi-LSTM instead of traditional LSTM is that, in application scenarios where GM-LSTM is employed for ZTD estimation or tropospheric correction, data from ERA5 and $VMF3$ are 24-hour knowable, no matter from reanalysis products or forecast products. This means that when inferring the ZWD at time t, $\{ZWD_{ERA5}, ZWD_{VMF3}\}$ from 00:00 to t as well as from $t+1$ to 23:00 are accessible for input. This allows Bi-LSTM to fully utilize the contextual information available at time t to extract features, particularly in scenarios with rapid water vapor changes, such as during rainfall.

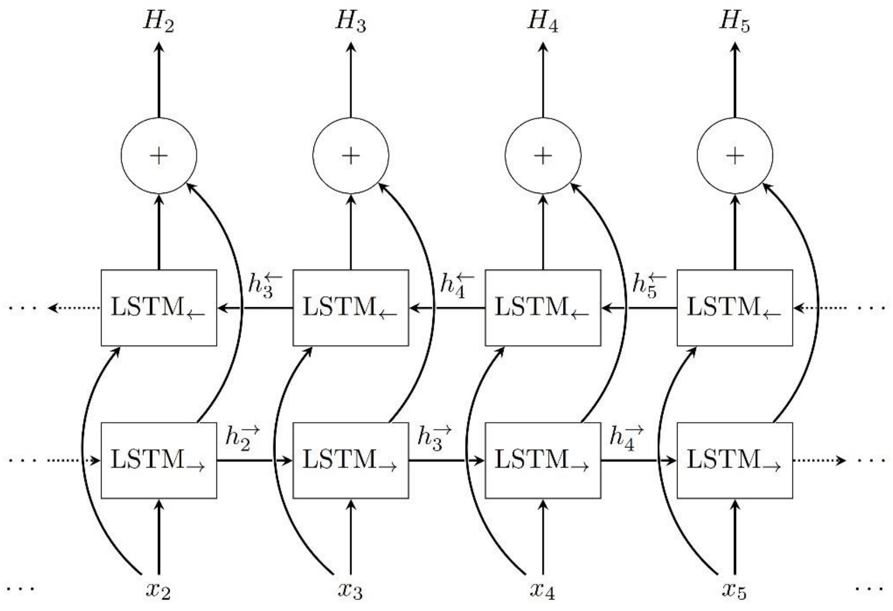


Fig. 5.5 Architecture of Bi-LSTM unit. The LSTM rectangle with a right arrow and a left arrow denotes the forward LSTM unit and backward LSTM unit, $h_t^{\rightarrow}$ and $h_t^{\leftarrow}$ are their output hidden state.

In the implementation of this study, each LSTM unit is configured with 2048 neurons, meaning that each LSTM unit outputs a 2048-dimensional feature vector at each hour. Specifically, a dropout probability of 0.2 is used to randomly mask a portion of the hidden states, effectively setting them to zero. This dropout regularization strategy helps mitigate overfitting by introducing stochasticity into the training process, encouraging the network to learn more robust and generalizable features from the input data. After the concatenate operation, each Bi-LSTM layer can transfer its input into a 4096-dimensional feature vector H_t, which includes its temporal context for reconstructing the probability density distribution of ZWD. The feature extraction process is described in Eqs. 5.4-5.6.

$$H_t^{(1)} = f_{BiLSTM}^{(1)}\left(X_t \middle| {h_{t-1}^{\rightarrow}}^{(1)}, {h_{t+1}^{\leftarrow}}^{(1)}\right), \tag{5.4}$$

$$H_t^{(2)} = f_{BiLSTM}^{(2)}\left(H_t^{(1)} \middle| {h_{t-1}^{\rightarrow}}^{(2)}, {h_{t+1}^{\leftarrow}}^{(2)}\right), \tag{5.5}$$

$$H_t^{(3)} = f_{BiLSTM}^{(3)}\left(H_t^{(2)} \middle| {h_{t-1}^{\rightarrow}}^{(3)}, {h_{t+1}^{\leftarrow}}^{(3)}\right), \tag{5.6}$$

where $f_{BiLSTM}^{(i)}$ denotes the i-th Bi-LSTM layer, $H_t^{(i)}$ is the output of the corresponding Bi-LSTM layer under its forward hidden state ${h_{t-1}^{\rightarrow}}^{(i)}$ and backward hidden state ${h_{t+1}^{\leftarrow}}^{(i)}$ with given input. $H_t^{(3)}$ is the final extracted feature, which includes the extracted context information of input X_t.

Since the probability distribution of ZWD is unknown, its pdf can be approximated using a Gaussian mixture model (GMM). The GMM consists of K Gaussian components, each characterized by independent parameters: mixing coefficients α, mean μ, and variance σ^2. Unlike traditional GMM constructed using the EM algorithm, those three parameters in this study are obtained by directly converting the 4096-dimensional feature vector $H_t^{(3)}$ through three different MLPs separately, as shown in Eqs. 5.7-5.9.

$$O_{\alpha;t} = MLP_{\alpha}\left(H_t^{(3)}\right), \tag{5.7}$$

$$O_{\mu;t} = MLP_{\mu}\left(H_t^{(3)}\right), \tag{5.8}$$

$$O_{\sigma^2;t} = MLP_{\sigma^2}\left(H_t^{(3)}\right), \tag{5.9}$$

where MLP_α, MLP_μ, and MLP_{σ^2} are three different MLP for convert $H_t^{(3)}$ into three K-dimensional vector $O_{\alpha;t}$, $O_{\mu;t}$, and $O_{\sigma^2;t}$. This approach enables this GMM to be directly output by the neural network in an end-to-end manner. The mean μ_i of each Gaussian component in the GMM is not subject to any constraints, denotes the components of $O_{\mu;t}$ as $o_{\mu_i;t}$, it can be directly used as $\mu_{i;t} = o_{\mu_i;t}$. However, to ensure that the cumulative distribution function is normalized, and the variance σ^2 is non-negative, additional constraints must be applied to the corresponding MLPs to prevent pathological solutions. To ensure that the mixing coefficients satisfy the normalization constraint $\sum_{i=1}^{K} \alpha_{i;t} = 1$, a SoftMax layer has been added after the MLP_α, which is used for generating mixing coefficients α, as shown in Eq. 5.10.

$$\alpha_{i;t} = SoftMax\left(o_{\alpha_i;t}\right) = \frac{e^{o_{\alpha_i;t}}}{\sum_{k=1}^{K} e^{o_{\alpha_k;t}}}, \tag{5.10}$$

where $o_{\alpha_i;t}$ is the i-th component of $O_{\alpha;t}$. The SoftMax function normalizes the output values to produce a probability distribution, ensuring that the mixing coefficients sum up to one.

To ensure that the variance is strictly positive and to avoid pathological configurations where the variance tends to zero, according to Bishop's work (Wicaksana & Rachman, 2018), assuming that $O_{\sigma^2;t}$ obeys a uniform distribution, using an exponential function to activate it will correspond to the choice of uninformative Bayesian priors. However, in the experiments of this research, it has been observed that during the initial stages of network training, the significant scale difference in ZWD values can cause certain components of the Gaussian mixture to diverge significantly from the target distribution. Consequently, errors propagated from the variance term become strongly amplified, resulting in exponential increases in gradients, eventually leading to gradient explosion and the subsequent collapse of the training process. While employing the ReLU activation function can mitigate gradient explosion, it indiscriminately suppresses all components of $O_{\sigma^2;t}$ negative to zero, potentially resulting in certain components of the standard deviation vector being forced to zero. This leads to the collapse of the pathologically configured mode. To address this challenge, this research proposes the adoption of a modified activation function known as the Piecewise Exponential Linear Unit (PELU), as demonstrated in Eq. 5.11.

$$\sigma_{i;t}^2 = PELU\left(o_{\sigma_i^2;t}\right) = \begin{cases} o_{\sigma_i^2;t} & o_{\sigma_i^2;t} > 1 \\ e^{o_{\sigma_i^2;t}} - 1 + 10^{-8} & o_{\sigma_i^2;t} \leq 1 \end{cases} \tag{5.11}$$

This modification aims to prevent gradient explosion while ensuring that the exponential behavior and enforced positivity of the standard deviation values are maintained as much as possible.

While the vector μ does not necessitate any explicit constraints, it has been found that MLP_μ exhibits sensitivity to gradients. Improper initialization can precipitate a gradient explosion, resulting in training failure or mode collapse, particularly during the early stages of training. As a solution, this research proposes initializing the bias term of MLP_μ based on the quantiles of ground truth Y. The quantile points are uniformly distributed within 0 to 1 according to the number of Gaussian combinations K.

2) Loss Function

The key to the proposed method is to consider ZWD as a conditional probability distribution rather than a numerical value. Consequently, conventional regression loss functions, such as the RMSE, cannot be directly employed for network training. To learn the pdf of ZWD from the ground truth Y, we employed the Maximum Likelihood Estimation (MLE) method to determine the optimal weights of GM-LSTM. Denoting $\boldsymbol{W}_{GM-LSTM}$ as the weights of GM-LSTM, the pdf of the GMM inferred by GM-LSTM at time t can be written as:

$$pdf_t(X_t|\boldsymbol{W}_{GM-LSTM}) = \sum_{i=1}^{K} \alpha_{i;t}(X_t|\boldsymbol{W}_{GM-LSTM}) \frac{1}{\sqrt{2\pi}\sigma_{i;t}(X_t|\boldsymbol{W}_{GM-LSTM})} exp\left(-\frac{\left(X_t - \mu_{i;t}(X_t|\boldsymbol{W}_{GM-LSTM})\right)^2}{2\sigma_{i;t}^2(X_t|\boldsymbol{W}_{GM-LSTM})}\right). \quad (5.12)$$

Consider minimizing the negative log-likelihood function is equivalent to maximizing the likelihood function, so the negative log-likelihood function of X_t can be used as the distribution part of the loss function at time t. Besides, each Gaussian component has been expected that it can describe its characteristics in a smaller ZWD space as much as possible to reduce the uncertainty of the model, so a penalty term $\sum_{i=1}^{K} \alpha_{i;t}\sigma_{i;t}^2$ has been added to the loss function to constrain each Gaussian component to be as narrow as possible. Hence, the loss at time t is:

$$Loss_t = \sum_{i=1}^{K} \alpha_{i;t}\sigma_{i;t}^2 - lnL(Y_t|X_t, \boldsymbol{W}_{GM-LSTM}) = \sum_{i=1}^{K} \alpha_{i;t}\sigma_{i;t}^2 - \sum_{j=1}^{N} \ln\left\{\sum_{i=1}^{K} \alpha_{i,j;t} \frac{1}{\sqrt{2\pi}\sigma_{i,j;t}} \exp\left(-\frac{\left(Y_{t,j;t} - \mu_{i,j;t}\right)^2}{2\sigma_{i,j;t}^2}\right)\right\}, \quad (5.13)$$

where $L(Y_t|X_t, \theta)$ is the likelihood function of Eq. 5.12, and N is the number of samples of X, i.e. the number of days available for the samples. And the total loss is the sum up of $Loss_t$ for each hour of the day, i.e.:

$$Loss = \sum_{t=0}^{23} Loss_t. \quad (5.14)$$

5.1.3.3 Training Strategy and Inference

After initializing the network and configuring the loss function, the GM-LSTM can be trained through BPTT algorithm described in section 4.4. Based on the experimental results of this study, it is recommended to use 5 Gaussian components (K = 5) to construct the pdf of ZWD. Experiments show that 5 Gaussian components are enough to identify ZWD patterns affected by extreme weather. To mitigate the risk of overfitting, it is advisable to employ the subsequent methodology for training the GM-LSTM model: iterative optimization utilizing the Adam optimizer over a span of 10,000 iterations, commencing with an initial learning rate set at 0.001. Furthermore, for every 500 iterations, the learning rate is adjusted by incorporating an exponential decay with a decay factor of 0.99.

Upon the completion of training, the acquired pdf can serve to infer the corrected ZWD, denoted as $ZWD_{GM-LSTM}$, at any location within the study area. This involves inputting a 24-hour $\{ZWD_{ERA5}, ZWD_{VMF3}\}$ series of that location into the trained GM-LSTM network and conducting forward propagation without dropout, thereby obtaining a GMM at that location. Given the absence of a priori knowledge regarding the $ZWD_{GM-LSTM}$, the inferred value is determined by the mean m_t of the mixture model, shown in Eq. 5.15. This is equivalent to the least squares estimate of independent sampling each component of the mixture model.

$$ZWD_{GM-LSTM;t} = m_t = \sum_{i=1}^{K} \alpha_{i;t}\mu_{i;t}. \quad (5.15)$$

Then the inferred ZTD at time t is:

$$ZTD_{GM-LSTM;t} = ZWD_{GM-LSTM;t} + ZHD_{VMF3;t.} \tag{5.16}$$

Simultaneously, the uncertainty associated with the inference can be gauged by the standard deviation of the mixture model as shown in Eq. 5.17:

$$s_t = \sqrt{\sum_{i=1}^{K} \alpha_{i;t} \left[\sigma_{i;t}^2 + \left(\mu_{i;t} - m_t\right)^2\right]}. \tag{5.17}$$

5.1.4. Data and Experiments

In this research, 8 areas in Europe with different latitudes are selected to conduct experiments to evaluate the performance of the proposed method in different regions. There are Portugal-Beja, Spain-Burgos, France-Paris, France-Brive-la-Gaillarde, German-Tübingen, Netherland-Groningen, Sweden-Sala, and Sweden-Svappavaara. For each area, several GNSS stations within that area from NGL products are prepared, and one of them was selected (usually near the center) as the test station and others were used for training. Those test stations did not participate in training and were completely unknown to GM-LSTM, so they could be used as a reference to evaluate the performance of GM-LSTM. Considering the seasonal variability of ZWD, for each region, the experiments tested the performance of the proposed method separately for 360 consecutive hours in winter (Nov. 26 to Dec. 10, 2021) and summer (Jun. 25 to Jul. 09, 2022), except for France-Paris. Since the time integrity of the GNSS stations near France-Paris from Jun. 25 to Jul. 09, 2022, was too bad and the loss of observations throughout the whole day occurred, the summer dates of the test for France-Paris have been changed to Jul. 09 to Jul. 23, 2021. The details of the test stations are presented in Table 5.1, while the location of all the training stations from NGL has been illustrated in Fig. 5.7, and the information of those stations can be accessed at https://github.com/hgwxx1945/DeepZTD/blob/main/STATION_INFO.txt.

Table 5.1 The details of GNSS stations for testing

Region	**Station ID**	**Latitude (°)**	**Longitude (°)**	**Height (m)**
Beja	MESS	37.835	-8.245	258.89
Burgos	BURG	42.347	-3.687	939.60
Paris	OP71	48.836	2.335	124.58
Brive-la-Gaillarde	BRIV	45.157	1.488	158.14
Tübingen	D400	48.519	9.078	382.17
Groningen	ENGE	53.213	6.644	42.43
Sala	0GRA	60.071	14.980	357.25
Svappavaara	0SVP	67.649	21.055	381.81

The proposed method has been evaluated using the metrics RMSE, Mean Bias Error (MBE), and Standard Error (SE), as shown in Eqs 5.18-5.21, compared with the conventional benchmarks of ERA5 ray tracing, VMF3 troposphere products, and GACOS[6] ZTD products

[6] Data can be accessed at: http://www.gacos.net/

(Yu, et al., 2018) and deep learning benchmark DNN across eight regions above. The metrics evaluation results are presented in Table 5.2, and the errors at each hour ΔZTD_t are illustrated in Fig. 5.6.

$$\Delta ZTD_t = ZTD_{GNSS;t} - ZTD_{model;t}, \tag{5.18}$$

$$RMSE = \sqrt{\frac{\sum_t \Delta ZTD_t{}^2}{N}}, \tag{5.19}$$

$$MBE = \frac{\sum_t \Delta ZTD_t}{N}, \tag{5.20}$$

$$SE = \sqrt{\frac{\sum_t (\Delta ZTD_t - MBE)^2}{N}}, \tag{5.21}$$

where $ZTD_{model;t}$ is the ZTD generated by the model needs to be evaluated at time t, and N is the total number of hourly acquisitions for that test station.

It is worth noting that since the configuration of DNN used in Osah et al. (2021), Yang et al. (2021a) and Yang et al. (2021b) may be relatively simple and cannot be used as a fair benchmark for DNN, this study further improved the performance of the DNN method for fair comparison. Specifically, by increasing the depth and number of neurons in DNN, the performance limits of DNN in this task were reached. Grid search showed no significant performance improvement when the DNN depth exceeded 8 layers with 256 neurons per layer, so an 8-layer DNN with 256 neurons per layer was used for comparison—much deeper and performing better than the 4-layer, 200-neurons in Osah et al. (2021), 2-layer, 5-neurons in Yang et al. (2021a), and 2-layer, 4-neurons in Yang et al. (2021b). This enhanced DNN configuration serves as a more rigorous deep learning benchmark for evaluating the proposed GM-LSTM model.

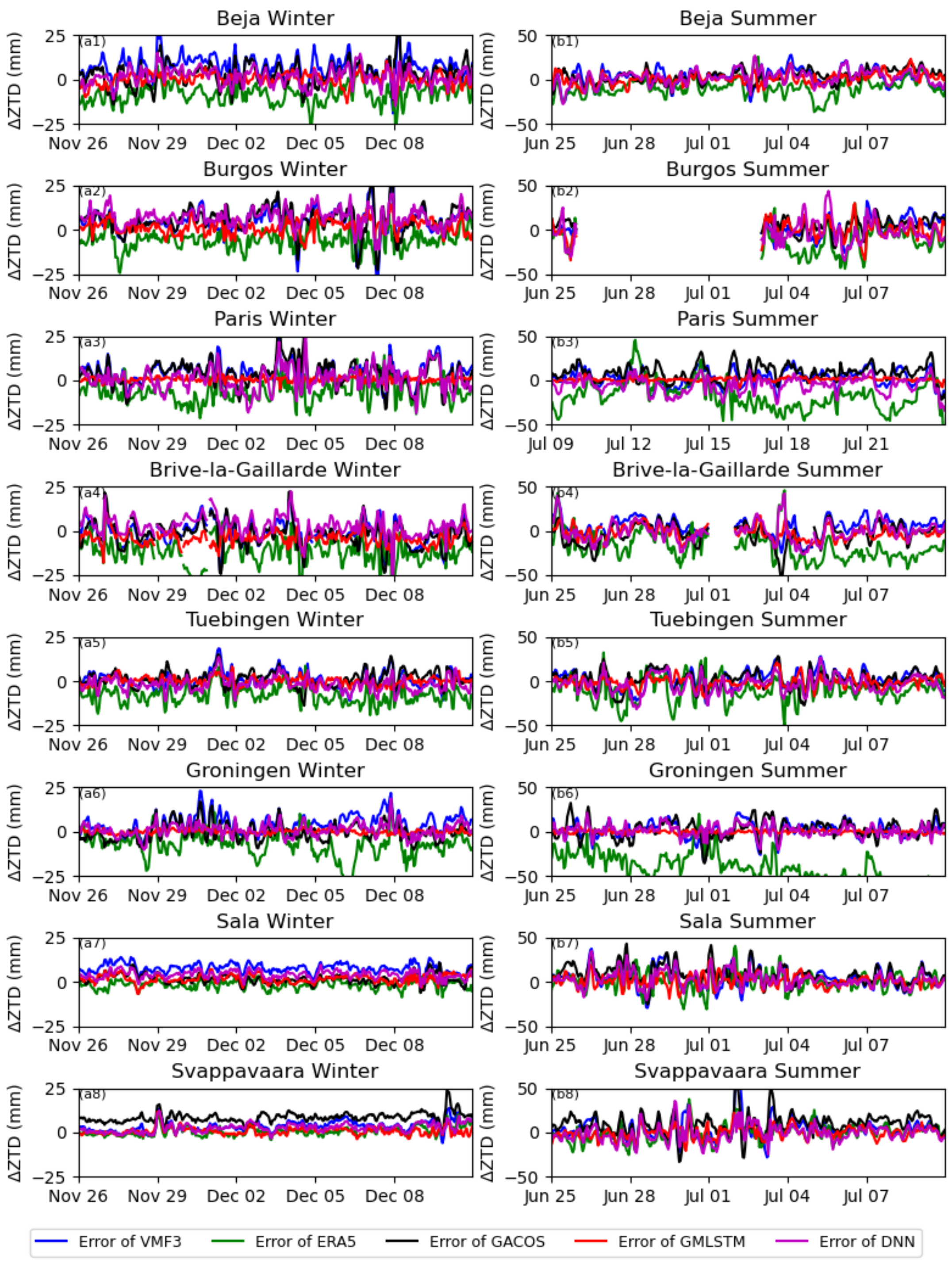


Fig. 5.6 The error of ZTD from DNN, VMF3, ERA5, GACOS, and GM-LSTM at eight testing areas. (a1) - (a8) represents the error from wintertime and (b1) - (b8) represent the error from summertime. Since the observation files from Jun. 26 to Jul. 2, 2022 at Burgos and Jul. 1, 2022 at Brive-la-Gaillarde are missing, the results during those times are not used to evaluate the model performance.

Table 5.2 The evaluation results at eight test stations

Region	Season	GM-LSTM			DNN			ERA5			VMF3			GACOS		
		RMSE (mm)	MBE (mm)	SE (mm)	RMSE (mm)	MBE (mm)	SE (mm)	RMSE (mm)	MBE (mm)	SE (mm)	RMSE (mm)	MBE (mm)	SE (mm)	RMSE (mm)	MBE (mm)	SE (mm)
Beja	Winter	**3.53**	**0.16**	**3.53**	5.00	1.36	4.81	9.78	-8.23	5.29	9.48	7.16	6.22	7.22	3.19	6.47
	Summer	**6.61**	**1.11**	**6.52**	8.47	1.93	8.25	13.19	-8.90	9.73	9.51	2.84	9.07	8.23	3.58	7.42
Burgos	Winter	**4.51**	**1.86**	**4.11**	8.84	7.16	5.19	7.76	-5.53	5.44	8.71	4.79	7.27	9.33	5.80	7.31
	Summer	10.22	**-0.47**	10.21	14.04	-1.93	13.91	19.16	-12.23	14.74	11.68	1.44	11.59	**10.03**	5.82	**8.17**
Paris	Winter	**1.53**	**0.54**	**1.43**	6.84	0.73	6.80	8.68	-6.20	6.08	7.43	4.42	5.97	8.13	3.81	7.18
	Summer	**2.29**	**1.08**	**2.02**	10.57	-5.76	8.86	24.03	-17.64	16.31	9.85	2.86	9.42	12.42	8.46	9.10
Brive-la-Gaillarde	Winter	**5.69**	-4.18	**3.85**	6.46	2.88	5.79	14.37	-12.09	7.77	7.10	**-1.54**	6.93	7.47	-1.96	7.20
	Summer	**7.74**	-2.40	**7.36**	11.73	**-0.96**	11.69	23.44	-17.31	15.80	11.05	4.78	9.96	12.68	-4.47	11.87
Tübingen	Winter	**2.37**	**0.06**	**2.37**	4.31	-1.71	3.95	9.22	-7.80	4.91	4.66	1.05	4.54	4.54	1.52	4.28
	Summer	**6.19**	**-0.93**	**6.12**	10.33	-3.39	9.76	16.42	-9.66	13.29	10.45	2.08	10.24	10.12	2.27	9.86
Groningen	Winter	**1.14**	**0.00**	**1.14**	3.86	0.95	3.74	10.02	-6.99	7.18	6.87	4.38	5.29	4.93	-0.18	4.93
	Summer	**1.76**	**-0.09**	**1.76**	6.70	1.07	6.61	46.10	-42.20	18.55	9.48	1.98	9.27	10.19	2.61	9.86
Sala	Winter	2.95	1.83	2.32	4.71	4.16	**2.20**	**2.79**	**-1.47**	2.37	8.25	7.77	2.78	3.34	1.87	2.76
	Summer	**6.84**	**1.51**	**6.67**	9.53	3.23	8.96	10.66	1.59	10.54	11.95	4.50	11.08	13.60	8.63	10.51
Svappavaara	Winter	**2.22**	0.70	**2.10**	3.56	2.59	2.45	2.52	**0.61**	2.44	4.59	3.57	2.89	8.92	8.47	2.81
	Summer	**6.81**	**-0.04**	**6.81**	9.41	-1.07	9.35	9.52	-0.27	9.52	12.48	4.14	11.78	14.89	9.55	11.43

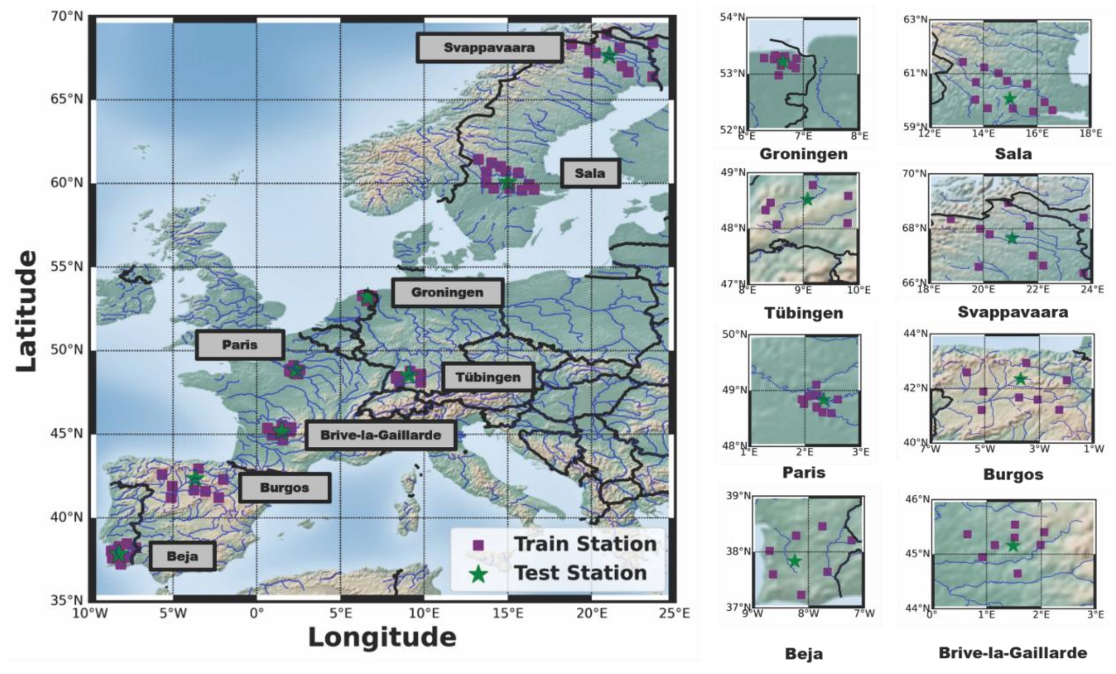


Fig. 5.7 The overview of selected GNSS stations from NGL

5.1.5. Discussion

5.1.5.1 Performance Evaluation

The average RMSE of ZTD retrieved by GM-LSTM, DNN, ERA5, VMF3, and GACOS at 8 test stations were 4.52 mm, 7.77 mm, 14.23 mm, 8.97 mm, and 9.13 mm, which demonstrated that the GM-LSTM model, once trained, was capable of providing ZTD estimates that achieve the current state-of-the-art level. Compared with DNN, ERA5, VMF3, and GACOS, the proposed GM-LSTM achieves an average RMSE reduction calculated by Eq.5.22 of 41.78%, 68.20%, 49.56%, and 50.43%. In the wintertime when the water vapor is relatively stable, the average RMSE of ZTD retrieved by GM-LSTM, DNN, ERA5, VMF3, and GACOS at 8 test stations are 2.99 mm, 5.45 mm, 8.14 mm, 7.14 mm, and 6.73 mm respectively, while in summer when water vapor is more active, the average RMSE of those 5 models are 6.06 mm, 10.10 mm, 20.31 mm, 10.81 mm and 11.52 mm. This illustrated that the ZTD estimated error of GM-LSTM decreased by 45.10%, 63.26%, 58.09%, and 55.59% in winter compared with DNN, ERA5, VMF3, and GACOS, and by 39.99%, 70.18%, 43.93%, and 47.42% in summer.

$$\text{Relative Lift} = \frac{RMSE_{\text{compared model}} - RMSE_{GM-LSTM}}{RMSE_{\text{compared model}}} * 100\% \tag{5.22}$$

More importantly, the $ZTD_{GM-LSTM}$ values are close to unbiasedness. This property may be a key factor in the significantly enhanced performance of the GM-LSTM compared to other tropospheric models. The average bias of $ZTD_{GM-LSTM}$ was 0.05 mm across eight test stations, with a seasonal variation of 0.12 mm in winter and -0.03 mm in summer. This can be attributed to the ability of GM-LSTM to assimilate the correction pattern of ZWD_{GNSS}. On the contrary, the meteorological data derived through ERA5 interpolation may exhibit a certain error. The temperature part of this interpolation error tends to accumulate quadratically, resulting in a large bias and standard deviation of ZTD_{ERA5}. In our experiments, the ZTD_{ERA5} demonstrated an average bias of -9.65 mm and a standard deviation of 9.37 mm. This included an average bias of -5.96 mm, and a standard deviation of 5.19 mm during winter, as well as an average bias of -13.33 mm, and a standard deviation of 13.56 mm during summer. It suggests that the ZTD estimated by the ERA5 ray-tracing method was underestimated by an average of about 9 mm (approximately 5mm in winter and 13mm in summer), and is consistent with the 10-15mm biases from the ZTD obtained from the ERA5 10-15 mm biases from the ZTD obtained from the ERA5 ray-tracing method as reported in Liu et al. (2022) and Ding et al. (2023). Furthermore, the grid NWM, like ERA5, cannot accurately reflect local water vapor activities, such as local storms or rainfall. This limitation results in a significant underestimation of the estimated ZTD in coastal areas, particularly during the summertime. The magnitude of this error may surpass 50 mm, as illustrated in Fig. 5.6 (b3) and (b6).

For ZTD_{VMF3} and GACOS ZTD products, they were overestimated by about 3.5 and 3.7 mm, respectively. This overestimation was attributed not only to errors from the NWM but also to the low temporal resolution. The NWM, which VMF3 and GACOS used with a temporal resolution of 6 hours, may not accurately capture rapid changes in water vapor activity. In contrast, the GM-LSTM model achieves a higher temporal resolution of 1 hour by utilizing both ZWD_{ERA5} and ZWD_{VMF3} as input data. Furthermore, the GM-LSTM shares the same mapping coefficients as VMF3, which GACOS cannot use. This means it can achieve a more precise slant delay than GACOS if the application is needed.

As another deep learning approach, the average bias of ZTD_{DNN} is 0.7 mm. Although not as good as GM-LSTM, it is still much smaller than the rest of the traditional methods. This demonstrates the effectiveness of data-driven deep learning methods in ZTD estimation tasks. However, the DNN cannot capture temporal features, and numerical fitting of ZTD limits their capacity to detect the spatial heterogeneity of ZWD. This limitation results in an average SE of 9.67 mm of DNN across the eight test areas, which is comparable to the average SE of 9.78 mm for GACOS. While the DNN outperforms ERA5 and VMF3, which have average SE of 13.56 mm and 10.30 mm, respectively, its error remains substantially higher than that of the GM-LSTM model, which achieves an average SE of 5.93 mm. This discrepancy likely contributes to the relatively poor performance of DNN in ZTD estimation tasks when compared to the proposed GM-LSTM model.

5.1.5.2 Effective Range

Due to the ZTD_{GNSS} of nearby GNSS stations being used as supervisory data during training, the distance of the GNSS stations to the study area may influence the accuracy of the GM-

LSTM model. If the GNSS station is too far from the study area, the learned features may not accurately represent the atmospheric conditions in the study area, leading to potential inaccuracies in the model's predictions. Conversely, if the GNSS station is close to the study area, the learned features are likely to be more representative, potentially improving the model's accuracy.

To investigate the influence of distance on accuracy and validate the reliability of the proposed GM-LSTM method, the OP71 station in Paris was employed as a reference station for testing and examined the performance of five models trained by using ZTD_{GNSS} from NGL station groups within distance ranges from the reference station: 0-15 km, 15-50 km, 50-100 km, 100-150 km, and 150-200 km. And the metric relative lift, described in Eq. 5.22, has been used to measure the improvement efficiency of the proposed GM-LSTM model relative to the compared model, including ERA5, VMF3, and GACOS. The geographical positions of the training stations for these five scenarios are depicted in Fig. 5.8. Except for the different training data, the training processes of those five models are identical, and their performance comparison is shown in Table 5.3 and Fig. 5.9.

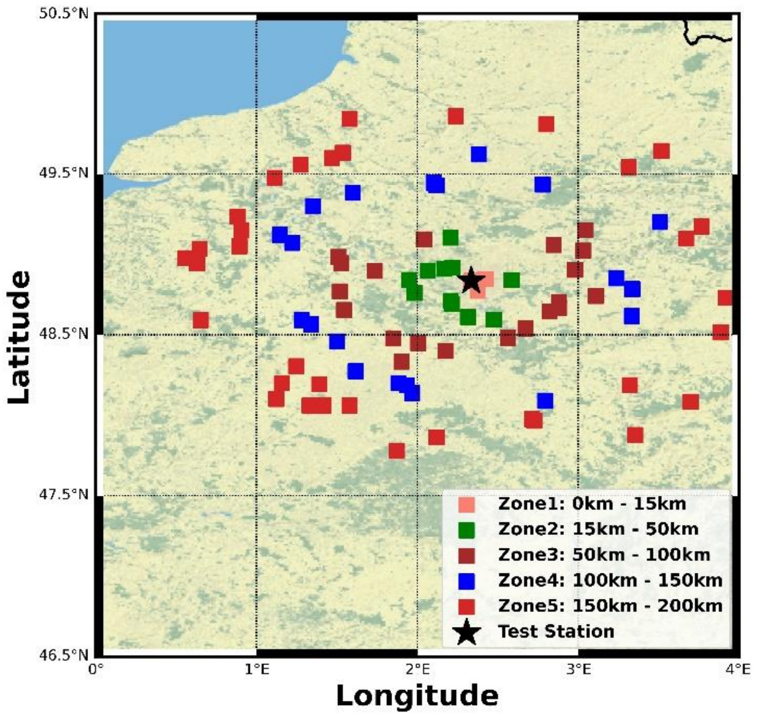


Fig. 5.8 NGL station groups with distances of 0-15 km, 15-50 km, 50-100 km, 100-150 km, and 150-200 km to OP71, Paris.

Table 5.3 The evaluation results of OP71 for five models trained by using different distance supervisory data

Range (km)	Season	RMSE (mm)	MBE (mm)	SE (mm)
0-15	Winter	1.70	0.66	1.57
	Summer	2.48	1.07	2.24
15-50	Winter	1.79	0.66	1.66
	Summer	2.77	0.43	2.74
50-100	Winter	2.31	0.41	2.27
	Summer	4.44	-0.04	4.44
100-150	Winter	3.48	-0.05	3.48
	Summer	6.24	0.23	6.23
150-200	Winter	3.88	-0.34	3.86
	Summer	6.17	-1.48	5.99

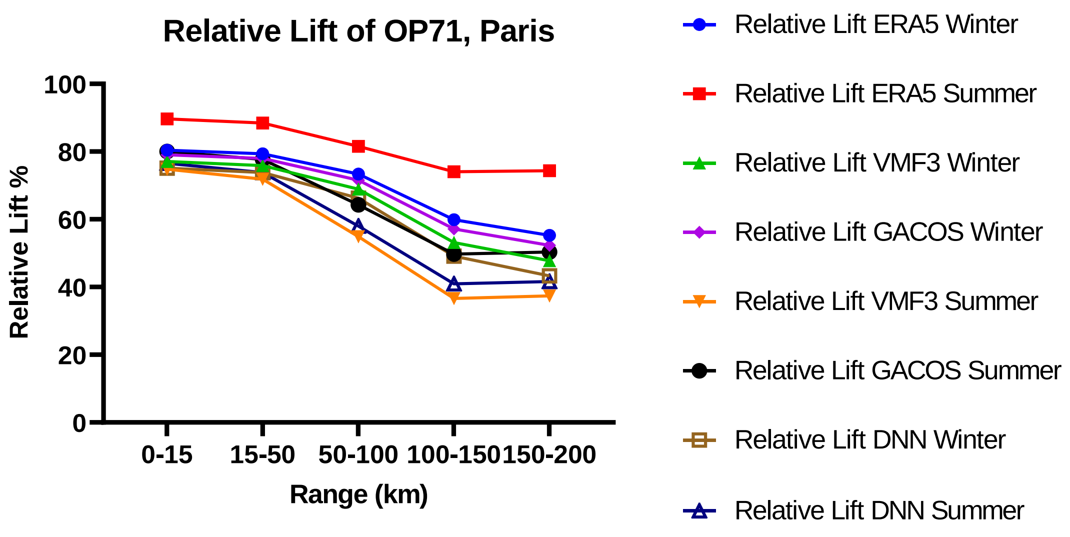


Fig. 5.9 Relative lift of OP71 between DNN, ERA5, VMF3, GACOS, and GM-LSTM in summer and winter.

As can be seen from Table 5.3 and Fig.5.9, the performance of the proposed GM-LSTM model decreases with the distance of the study area from the surrounding GNSS sites available for training. When GNSS stations are available for training within 50 km of the study area, the proposed GM-LSTM model can achieve a relatively stable performance improvement, nearly 80% improvement compared with other tropospheric models mentioned above. However, when the distance between the training stations and the study area is between 50 and 150 kilometers, its relative lift drops significantly as the distance increases. When the training stations are more than 150 kilometers away from the study area, distance no longer has a significant impact on model performance. At this time, GM-LSTM can obtain approximately 40% improvement compared to the best one of ERA5, VMF3, and GACOS. We believe that the reason for the performance decreases comes from the heterogeneity of the training data and tropospheric activity in the study area.

In addition, to explore the impact of geographical location on GM-LSTM performance, we conducted a Pearson correlation analysis and significance test on the RMSE and the longitude, latitude, height, and distance from the sea of the proposed GM-LSTM model across eight test stations, the results are shown in Fig. 5.10. These results indicate a statistically significant correlation between the RMSE of GM-LSTM and station altitude only in summer. This correlation likely arises from the increased uncertainty in ZWD due to high water vapor activity during the summer months. Lower-altitude areas may exhibit distinct ZWD patterns compared to higher-altitude regions, making accurate ZWD estimation at high altitudes challenging for tropospheric models. In fact, similar seasonal inaccuracies are observed in models like ERA5, VMF3, and GACOS, particularly in high-altitude areas during summer.

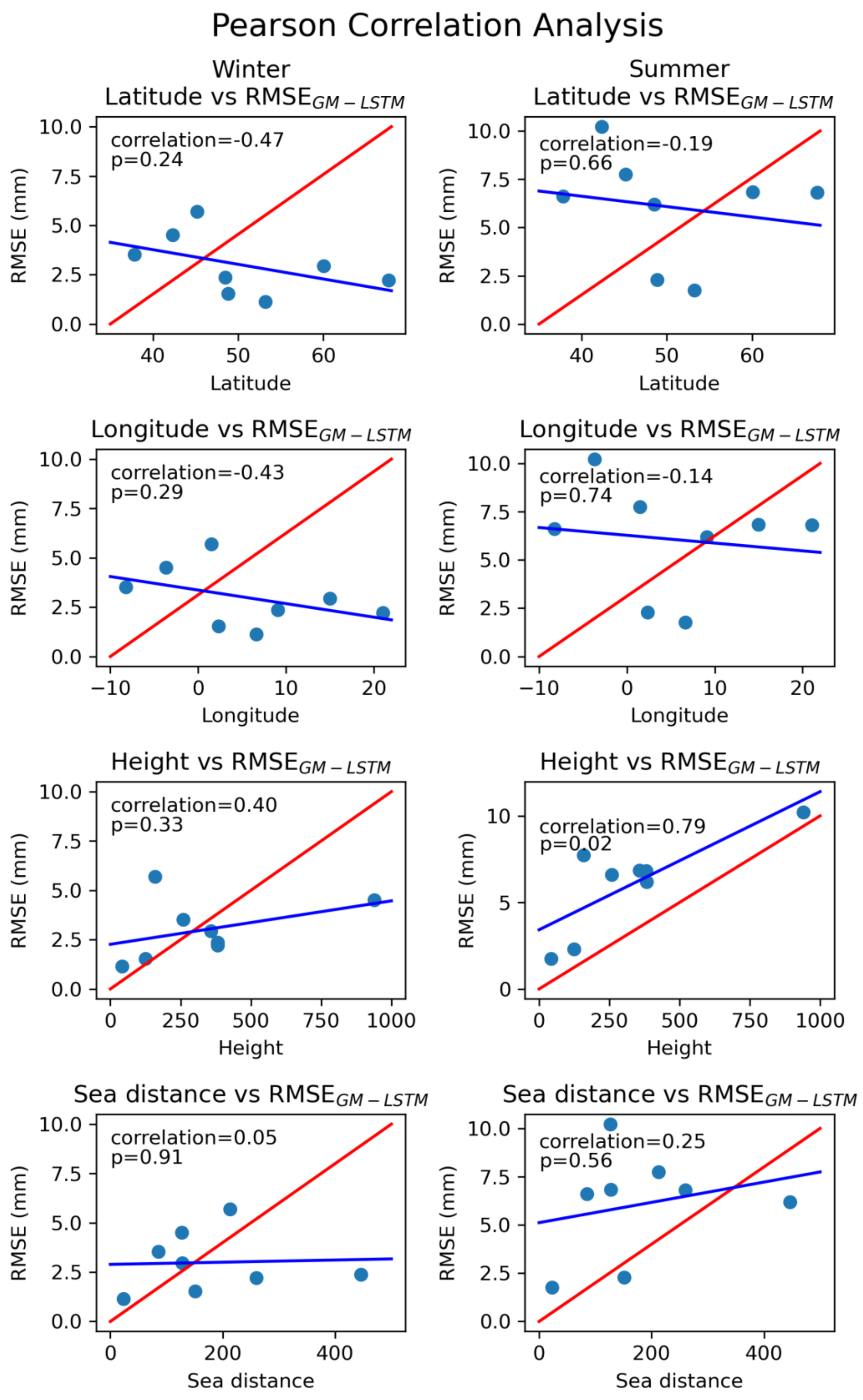


Fig. 5.10 Pearson correlation analysis and significance test on the RMSE of GM-LSTM and the longitude, latitude, height, and distance from the sea. The red line represents the complete correlation for reference, and the blue line represents the actual linear fit.

For the GM-LSTM model, the training process incorporates data from surrounding GNSS stations, which may have different elevations than the test stations. As a result, the model may not fully capture the ZWD correction pattern specific to the test altitude. Nonetheless, the GM-LSTM model remains independent of the longitude and latitude of the application area, outperforming traditional methods and DNN. This generalizability makes GM-LSTM a promising tropospheric model for broad geographic applications.

5.1.5.3 Model uncertainty and spatial heterogeneity of the tropospheric delay

As mentioned in section 5.1.2, GM-LSTM works under the assumption that the learned mapping patterns are shared within the local areas. For most test scenarios, this assumption can be satisfied. At that time, the standard deviations of GM-LSTM were small and tended to use a dominant Gaussian component to express the mapping with ZWD. However, since the GNSS stations for training around the study area have different locations and topography, and the ERA5 and VMF3 grid data cannot capture the impact of local meteorological activities, such as thunderstorms, rain, etc., there may be multiple ZWD mapping patterns in that local area. This spatial heterogeneity will cause GM-LSTM to use different Gaussian components to capture the information provided by GNSS stations with varying patterns of mapping. Besides, the ZWD distribution inferred from GM-LSTM presents a multimodal distribution and no longer has a dominant Gaussian component, which ultimately results in a Gaussian mixture exhibiting a larger standard deviation. In this case, GM-LSTM is capable of providing that the probability $P(a \leq ZWD_{GM-LSTM} \leq b)$ utilizing Eq. 5.23, as a measurement of uncertainty. The moments with the largest standard deviation among the eight test areas have been taken as an example to analyze the impact of meteorological activities on model uncertainty, as shown in Fig. 5.11.

$$P\left(a \leq ZWD_{GM-LSTM;t} \leq b\right) = \int_a^b \sum_{i=1}^{K} \alpha_{i;t} \frac{1}{\sqrt{2\pi}\sigma_{i;t}} exp\left(-\frac{\left(x-\mu_{i;t}\right)^2}{2\sigma_{i;t}^2}\right) dx. \qquad (5.23)$$

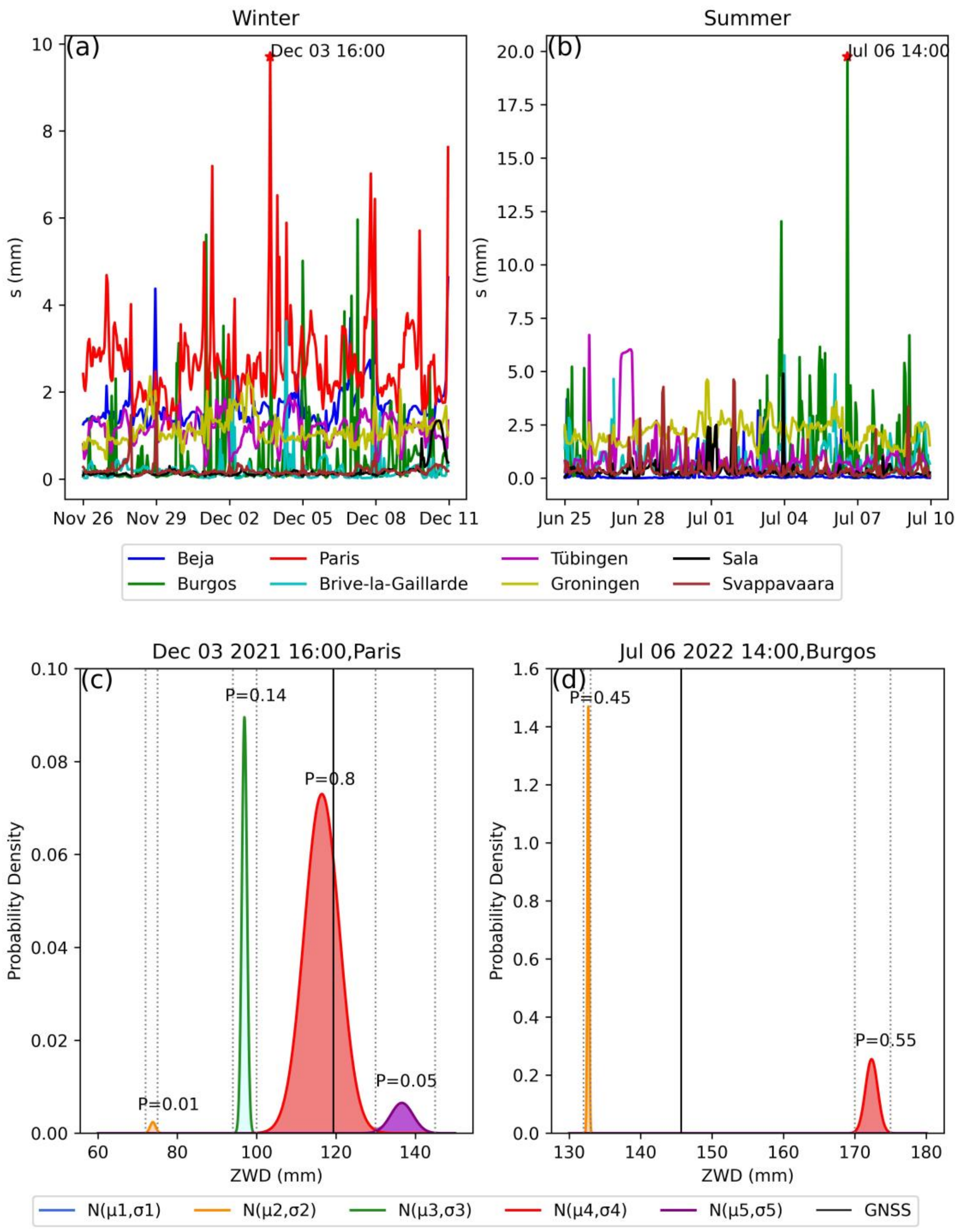


Fig. 5.11 (a) Standard deviation of the test areas from Nov. 26 to Dec. 11, 2021. (b) Standard deviation of the test area from Jun. 25 to Jul. 10, 2022. Please note that Paris is not shown in this subfigure due to different test dates. (c) The probability density generated by GM-LSTM in Paris at 16:00 on Dec. 8, 2021. (d) The probability density was generated by GM-LSTM in Burgos at 14:00 on Jul. 6, 2022.

From Nov. 26 to Dec. 10, 2021, the largest standard deviation of GM-LSTM inference was in Paris on Dec. 3, 16:00, which was 9.71 mm, as shown in Fig. 5.11 (a). At this time, GM-LSTM generated 4 effective Gaussian components and divided the probability space into:

$$P(72 \leq ZWD < 75) = 0.01$$

$$P(94 \leq ZWD < 100) = 0.14$$

$$P(100 \leq ZWD < 130) = 0.80$$

$$P(130 \leq ZWD < 145) = 0.05$$

And for the summertime, Jun. 25 to Jul. 09, 2022, the largest standard deviation inference was in Burgos on Jul. 6 14:00, which was 19.75 mm, shown in Fig. 5.11 (b). There were only two effective Gaussian components that separated the probability space into two parts, which are:

$$P(131 \leq ZWD < 133) = 0.45$$

$$P(170 \leq ZWD < 175) = 0.55$$

The reason for this uncertainty may be related to meteorological activity. According to the Meteorological Terminal Aviation Routine Weather Report (METAR) weather records from airports near Paris (Le Bourget Airport, Charles de Gaulle Airport, Orly Airport, Saclay-Versailles Airport, and Villacoublay Velizy Air Base), there was mist and drizzle in southeast Paris at 16:00 on Dec. 3, 2021[7]. However, Pontoise Cormeilles Airport, northwest of Paris, recorded no rainfall then. Similarly, the METAR weather record from Burgos Airport shows it was sunny from 12:00 to 14:30 on Jul. 6, 2022. However, according to the METAR weather record from Logrono Agoncillo Airport, about 115 km east of Burgos, and the severe weather record from the ESWD, there was heavy rain and thunderstorms during these times on the east of Burgos[8]. The E-OBS daily rainfall products[9] derived from on-site observations (Bandhauer et al., 2022) from European National Meteorological and Hydrological Services also confirmed this heavy rainfall, and its impact is shown in Fig. 5.12.

[7]Data can be accessed at https://www.ogimet.com/metars.phtml.en or https://mesonet.agron.iastate.edu/request/download.phtml

[8] Data can be accessed at https://eswd.eu/cgi-bin/eswd.cgi

[9] Data can be accessed at https://www.ecad.eu/download/ensembles/download.php

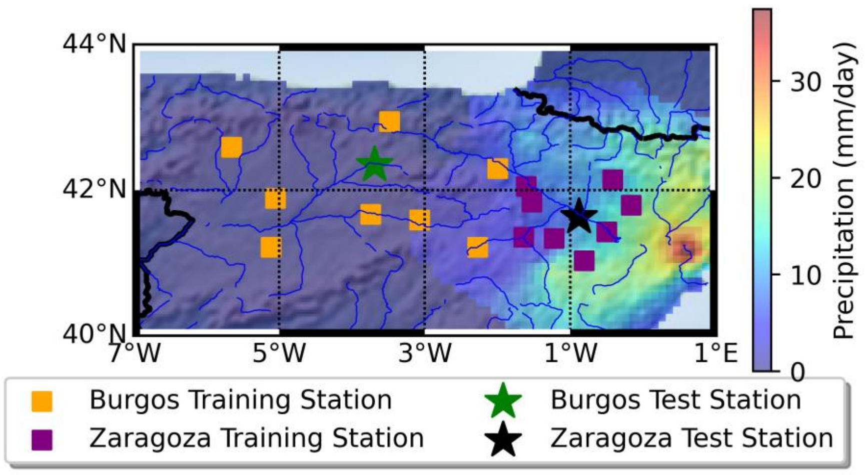


Fig. 5.12 The positions of NGL station groups around Burgos and Zaragoza and the daily precipitation amount on Jul. 6, 2022, retrieved from E-OBS.

Rainfall events will increase the integrated water vapor and thus increase the ZWD. When only part of the GNSS stations used for training are affected by rainfall events, those stations affected by rainfall will cause the network to produce Gaussian components with larger mean values than the stations not affected by rainfall. This will ultimately cause the GM-LSTM to present a multimodal distribution, and this multimodal distribution can provide not only the probability of ZWD inference but also the effective probability space of ZWD, which can be used to analyze the potential reason caused by that uncertainty. It is important to recognize that the uncertainties discussed previously emerge only under conditions of meteorological heterogeneity within the study region. Conversely, in the presence of uniform meteorological conditions across the study area, the GM-LSTM can deduce ZWD of superior quality, even amidst extreme weather events such as heavy rainfall. Regarding the extreme rainfall events reported near Zaragoza, the performance of GM-LSTM trained by homogeneous (within 50 km around Zaragoza) and heterogeneous (within 200 km around Burgos) data have been tested under heavy rainfall situations, respectively. The distribution of training stations is shown in Fig. 5.12, and the results are shown in Fig. 5.13. In the case of heterogeneity, as depicted in Fig. 5.13 (a) and (b), the standard deviation of GM-LSTM precisely pinpoints the interval impacted by the heavy rainfall (14:00). Besides, the ZTD inferred during this timeframe fails to closely mirror the variations in ZTD_{GNSS} . In contrast, with homogeneous training data, as demonstrated in Fig. 5.13 (c) and (d), the ZTD estimated by GM-LSTM aligns more closely with the GNSS observations, maintaining a standard deviation below 1 mm throughout the whole day.

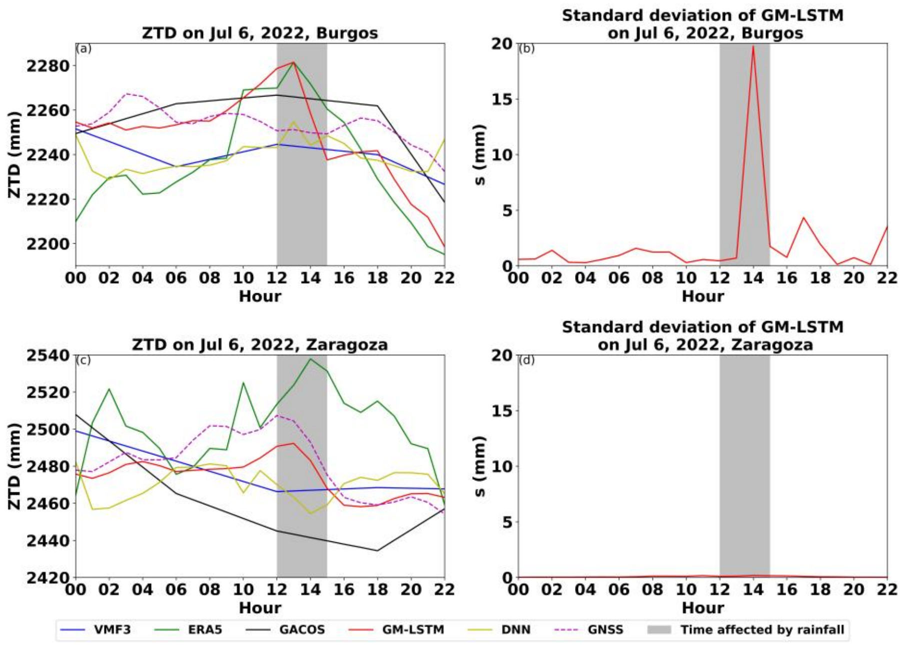


Fig. 5.13 Comparison of ZTD estimates of DNN, VMF3, ERA5, GM-LSTM, and GACOS under the influence of heavy rain. The grey background marks the time affected by rainfall (12:00-15:00) . (a) The ZTD estimations on Jul. 6, 2022, Burgos from the models above, where the GM-LSTM used was trained from the GNSS station within 200 km from GNSS station BURG in Burgos. (b) The standard deviation of GM-LSTM on Jul. 6, 2022, Burgos. (c) The ZTD estimations on Jul. 6, 2022, Zaragoza from the models above, where the GM-LSTM used was trained from the GNSS station within 50 km from GNSS station ZARA in Zaragoza. (d) The standard deviation of GM-LSTM on Jul. 6, 2022, Zaragoza.

The above experiments prove that the standard deviation of GM-LSTM can reflect the similarities of meteorological conditions between the test area and the training stations. When the standard deviation is large, model performance may decrease due to meteorological heterogeneity. In this case, it is recommended to use training stations closer to the test area to ensure that they have similar meteorological activities. When under extreme rainfall conditions, using GNSS stations with homogeneous meteorological conditions for training enables GM-LSTM to learn the observed tropospheric patterns by GNSS, thereby inferring accurate ZTD.

5.1.6. Conclusion

This research proposes a deep learning based tropospheric delay estimation method. By using a deep neural network called GM-LSTM, this approach allows the network to learn the mapping pattern from ZWD retrieved from ERA5 and VMF3 to the actually observed ZWD by GNSS. After training, the GM-LSTM is able to generate a series of Gaussian mixture probability densities of ZWD at any location within the study area and its corresponding probability distribution, which can be used to estimate ZWD precisely and evaluate the estimation uncertainty arising from weather activity.

After testing the performance of the method in winter and summer at eight different latitudes in Europe, compared with ZTD retrieved by DNN, ERA5, VMF3, and GACOS, the RMSE of ZTD inference by GM-LSTM was reduced by 41.78%, 68.20%, 49.56%, and 50.43% on average, achieving state-of-the-art performance. In addition, with the help of meteorologically homogeneous GNSS training data, the proposed method still performs well under heavy rain, which provides the ability to work in extreme weather that ERA5, VMF3, and GACOS do not have.

5.2. Real-Time GNSS Integrated Water Vapor Sensing Based on Time Series Correction Deep Learning Models

5.2.1. Background and problem statement

Water vapor is a significant component of the Earth's atmosphere in terms of energy transport by latent heat and radiative forcing. It has a vital role in the water and energy cycles (Worden et al., 2007), climate change (Karl & Trenberth, 2003), and the understanding of many extreme weather phenomena (Zhu & Newell, 1994). Due to the rapid spatiotemporal variations of water vapor, its real-time acquisition with high spatiotemporal resolution remains a challenge. Typically, water vapor can be quantified using integrated water vapor (IWV) in units of $\mathrm{kg/m^2}$, representing the mass of water vapor within a 1 $\mathrm{m^2}$ atmospheric column. Since the 1990s, with the development of satellite and remote sensing technologies, researchers have successfully retrieved IWV by employing high-spectral and multispectral remote sensing techniques, such as the Moderate Resolution Imaging Spectroradiometer (MODIS) (King et al., 1992), and atmospheric reanalyses (Schröder et al., 2018). However, these techniques still face limitations in water vapor estimation when it comes to being conducted in real-time, all-weather, and high accuracy. For example, multispectral remote sensing technologies like MODIS are susceptible to weather conditions, particularly cloud cover and diurnal changes (Vaquero-Martínez et al., 2017). Reanalysis data, such as ERA5 (Hersbach et al., 2023), can provide medium-resolution gridded data with a latency of about 5 days or more, limiting their availability for real-time or near real-time access. In addition, the traditional radiosondes, which utilize meteorological balloons, can offer high-precision vertical distribution of water vapor, but it is constrained by the limited number of measurements (Durre et al., 2009).

The basic principles of GNSS meteorology are introduced in Chapter 2 of this thesis, where Eqs. 2.55–2.57 demonstrate that the ZTD measured by GNSS can be converted to IWV, given the meteorological variable which is calculated by water vapor pressure and temperature from earth's surface to the tropopause (Yuan et al., 2021). Due to significant improvements in recent years, some GNSS stations can even provide real-time or near-real-time ZTD products. However, in the absence of professional meteorological equipment, obtaining T_m through real-time measurements is very challenging, which impedes the real-time retrieval of IWV. Besides, in real-time inversion applications, accurate NWM data products are also hard to obtain for calculate T_m. Additionally, some scenarios may lack network support, necessitating offline calculations. This underscores the need for a method to calculate T_m offline for GNSS real-time water vapor inversion. To obtain real-time weather data predictions, Landskron & Böhm (2018) put forward the GPT3 model by analysing ten years of mean monthly pressure level data from ERA-Interim products. GPT3 is composed of a series of spherical harmonic functions. By inputting the time and location information, it predicts meteorological data such as temperature, pressure, and water vapor pressure in an offline manner. The GPT3 model is widely acknowledged for its effectiveness; however, it exhibits limitations associated with geographical considerations, particularly in regions with high latitudes or that are experiencing significant climatic variations (Yang et al., 2021). Furthermore, due to its inherent design, the model fails to adequately account for diurnal fluctuations, hampering its ability to predict accurately the rapid and high-frequency variations inherent in diurnal meteorological data. To overcome this limitation, this section proposes a conversion model that uses a combination of DNN and LSTM to implicitly learn spatiotemporal patterns in meteorological data, allowing us to perform ZTD-to-IWV conversion without the need for actual meteorological observations but only GPT3 values. This is a combination, not a true combination, to cover different features in a different range of IWV values. Since LSTM-based learning can capture features from historical sequences of IWV, the method proposed in this paper is expected to mitigate high-frequency retrieval errors in ZTD-IWV inversion in the absence of current meteorological data.

5.2.2. Methodology

Due to the challenges of obtaining real-time and accurate meteorological data from GNSS stations, the proposed method uses a deep neural network f_{DNN} to implicitly learn meteorological information and directly map ZTD to IWV. According to Eqs. 2.55-2.57, IWV can be considered as a function of ZTD_{GNSS}, temperature T, water vapor pressure p_{w} and pressure p_s. Since T, p_{w} and p_s exhibit periodicity in temporal and spatial variations, these quantities can be modeled based on latitude LAT, longitude LON and height h_s as well as day of year DOY and hour of day HOD. Although not as accurate as ERA5, GPT3, as one of the latest empirical models, can map location and time information to meteorological variables offline. However, its error can be large compared to the IWV retrieved from ERA5, especially in mid- and high-latitude regions. In this study, the IWV retrieved from GPT3 IWV_{GPT3} has been used as part of the network training input. This will help learn from prior knowledge and achieve faster convergence. Therefore, as a numerical regression problem, a deep neural network f_{DNN} with weights $\boldsymbol{W}_{DNN}$ can be trained with the following input:

$$X = \{ZTD_{GNSS}, LAT, LON, h_s, DOY, HOD, T_{GPT3}, p_{s_{GPT3}}, p_{\mathrm{w}_{GPT3}}, IWV_{GPT3}\}, \tag{5.24}$$

and the ground truth Y is the IWV retrieved from ERA5:

$$Y = IWV_{ERA5}, \tag{5.25}$$

by using the RMSE as loss function, as described below:

$$Loss = \sqrt{\frac{1}{N}\sum (f_{DNN}(X; \boldsymbol{W}_{DNN}) - Y)^2}, \tag{5.26}$$

This network can be easily trained through BP algorithm described in section 4.3.

After training, f_{DNN} can be used offline, which has shown a significant improvement in accuracy compared to IWV_{GPT3}. However, as illustrated in Fig 5.14, the temporal correlation in ZWD results in a corresponding temporal correlation in the retrieved IWV, which cannot be effectively captured by DNN. To get more improvements, the limitations of $T_{GPT3}, p_{s_{GPT3}}$ and $p_{w_{GPT3}}$ need to be analyzed. From the difference between IWV_{GPT3} and IWV_{ERA5}, it can be found that the IWV retrieved error is related to its retrieved IWV value, that is, the water vapor activity level. Besides, when the IWV value is high (usually in summer, depending on the location), the correlation between retrieved IWV and the retrieved error of this IWV is low; on the contrary, when the IWV value is low (usually in winter and spring, depending on the location), the retrieved error is correlated with the IWV value. An example from KIRU station (GNSS station information is shown in Fig. 5.16 and Table 5.4) shows this property in Fig. 5.14. When the GNSS IWV is low, the IWV value retrieved using GPT3 has a strong linear correlation with its error (from January 1 to May 25, 2022, R^2=0.77). However, when the IWV value is high, this linear correlation declines rapidly (from January 1 to May 26 to August 4, 2022, R^2=0.13).

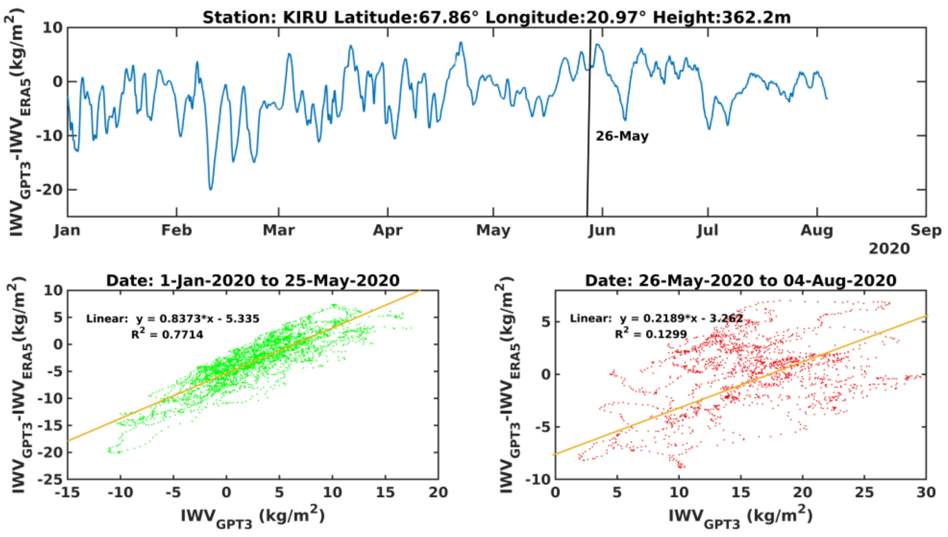


Fig. 5.14 The correlation between the IWV retrievals and its associated error. Before 26-May-2020, the R^2 between IWV and retrieved error is 0.77, while after is 0.13. Data from September to December 2020 at KIRU is not available because of missing observation files.

With this property, two different models can be trained separately for the active (High IWV value) and inactive periods (Low IWV value) of water vapor and used in combination to improve performance. The classification of high/low boundaries can be obtained through change point detection from Bayesian Estimator of Abrupt Change, Seasonality, and Trend (BEAST) (Zhao et al., 2019), or simply using empirical value. It is not necessary to find a very precise boundary because neural networks have the ability to generalize. For the case study in this research, 20 $\mathrm{kg/m^2}$ is a recommended threshold. The boundaries effect will be discussed in the next section.

As observed, DNN worked well with high-frequency water vapor changes, i.e., the high IWV part. Hence, for the high IWV part, it can be trained by a DNN by only using part of the training data that corresponds to high IWV value $X_{highIWV}$ and $Y_{highIWV}$. However, for the low IWV part, the short-term autocorrelation of water vapor variation with 3–5 hours lag is clearly observed; an example is shown in Fig.5.15, which allows the water vapor retrieval errors to be further corrected by time series models. Since the application scenario of real-time retrieval of IWV can only obtain input from the past, which differs from section 5.1, this study proposes using an LSTM network f_{LSTM} to model the time series for the low IWV part. Given that the lag of the ACF is 3–5 hours, indicating the presence of time features that could enhance the model within this period, the sequence length of the LSTM is set to 5 hours. Time series data are described using a sequence partial order relationship to represent temporal information. Hence, only GNSS ZTD and the IWV retrieved with GPT3 are taken as input features for training f_{LSTM}, as shown in Eq. 5.27.

$$X_{lowIWV;t} = \{ZTD_{GNSS;t}, IWV_{GPT3,t}\}, t = 1,2 \dots 5, \tag{5.27}$$

and the output $\hat{Y}_{lowIWV;t}$ is the predicted IWV time series with 5 consecutive hours, as described below:

$$\hat{Y}_{lowIWV;t} = f_{LSTM}(X_{lowIWV;t}). \tag{5.28}$$

The f_{LSTM} can be supervised by using the ground truth Y with the same time division by using the BPTT algorithm which is described in section 4.4.

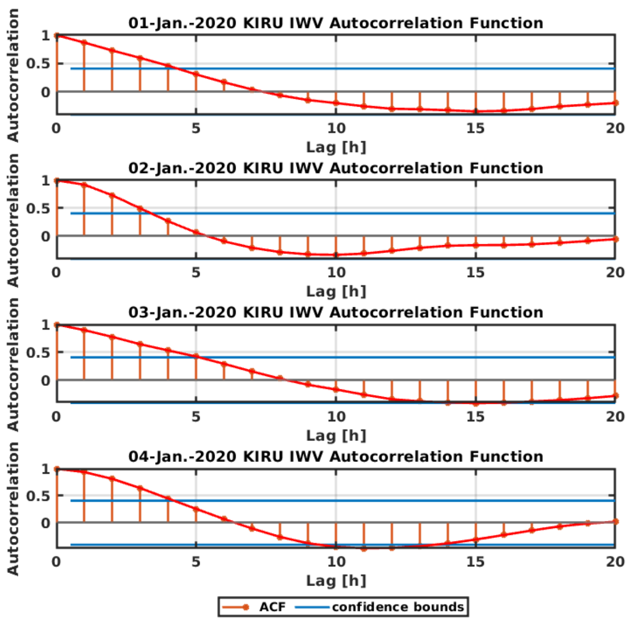


Fig. 5.15 ACF of diurnal GNSS IWV of KIRU, 01-Jan.-2020 to 04-Jan.-2020.

5.2.3. Data and experiments

There are four selected GNSS stations as the case study to validate the effectiveness of the proposed method, which are: KIRU in Kiruna, Sweden; DZYL and DZY1 in Delfzijl, Netherlands; and FFMJ in Frankfurt, Germany, to represent different climate types in the mid-to-high latitude regions of Europe. The details of those stations are described in Fig. 5.16 and Table 5.4.

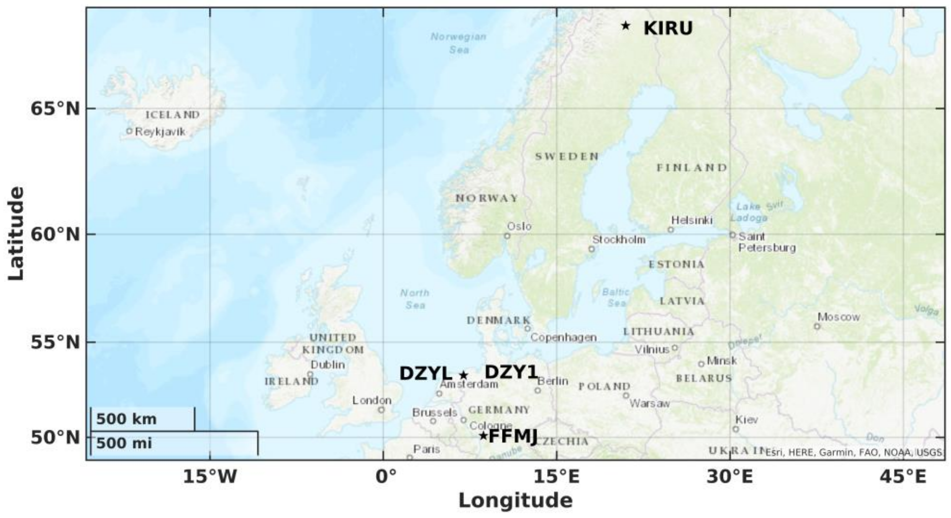


Fig. 5.16 The locations of GNSS stations KIRU, DZYL, DZY1, and FFMJ. Note that both DZYL and DZY1 are located in Delfzijl, Netherlands, and they are too close to be distinguished from the map.

Table 5.4 Coordinates of the GNSS stations selected for model validation.

GNSS Station	Longitude/ °	Latitude/ °	Height/ m
KIRU	20.9684	67.8574	362.2
DZYL	6.9404	53.3200	15.7
DZY1	6.9360	53.3228	8.2
FFMJ	8.6650	50.0906	130.1

Data from 2016 to 2019 were used for training, whereas the data in 2020 were used as an independent test set. Various metrics are used in the evaluation, such as RMSE, MBE, Mean Absolute Percentage Error (MAPE), and coefficient of determination (R^2):

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum\left(IWV_{pre} - IWV_{ERA5}\right)^2}, \tag{5.29}$$

$$MBE = \frac{1}{N}\sum(IWV_{pre} - IWV_{ERA5}), \tag{5.30}$$

$$MAPE = \frac{1}{N}\sum\left|\frac{IWV_{pre} - IWV_{ERA5}}{IWV_{ERA5}}\right| * 100\%, \tag{5.31}$$

$$R^2 = \frac{[\sum(IWV_{pre} - \overline{IWV_{pre}}) * (IWV_{ERA5} - \overline{IWV_{ERA5}})]^2}{\sum(IWV_{pre} - \overline{IWV_{pre}})^2 * (IWV_{ERA5} - \overline{IWV_{ERA5}})^2}, \tag{5.32}$$

where IWV_{pre} is the predicted IWV, and IWV_{ERA5} is the ground truth.

The proposed model is trained station by station. For each station, 90% of its data from 2016 to 2019 is randomly selected for training, and the remaining part from 2016 to 2019 is used as a validation set for parameter adjustment to select the model with the best generalization ability. To evaluate the performance of different models in different water vapor periods, GPT3, the DNN only, and the combined model with DNN and LSTM are compared. In the combined model, a DNN model was developed for high IWV values, whereas an LSTM model was used for low IWV values. In DNN only model and the one in the combined model, their network structures are the same, with 4 layers with 200 neurons in each layer. Training started with an initial learning rate of 0.01 and decayed by a factor of 0.7 every 500 epochs until it reached the max of 3000 epochs. For the LSTM part, it had been constructed with 256 neurons for each gate, and the learning strategy was: Initial learning rate 0.001, decay by a factor 0.8 every 500 epochs until it reached the max epoch 3000. To evaluate our approach, we compared the matrices RMSE, MBE, MAPE and R^2 between the proposed approach and GPT3, which are shown in Table 5.5. Besides, the RMSE for the entire year, high IWV period, and low IWV period are evaluated as shown in Fig. 5.17.

From the evaluation results, it can be observed that the combined model proposed in this research consistently achieves smaller RMSE compared to GPT3 and the DNN at all times. For the four GNSS stations KIRU, DZYL, DZY1, and FFMJ, the RMSE of the combined model is 2.24, 3.20, 3.15, and 2.78 $\mathrm{kg/m^2}$, respectively, representing improvements of 55%, 17%, 18%, and 15% compared to the GNSS IWV retrieved from GPT3. Since the proposed model is trained station by station, similar results were obtained at the DZYL and DZY1 stations located on both sides of the EMS canal to prove the stability of the proposed model. The improvement in retrieval performance of those four stations can mainly be attributed to the use of LSTM for IWV sequence correction during periods of low IWV. Compared to the GNSS IWV retrieved from GPT3, the combined model proposed in this research achieves retrieval improvements of 63%, 17%, 22%, and 14% during low IWV periods, significantly enhancing IWV retrieval in the Northern European region. In fact, Ding & Chen (2020) have indicated that the stability of GPT3 model predictions for temperature and pressure decreases with increasing latitude. During periods with low IWV in high-latitude regions, where water vapor variations are not drastic, the substantial distortions in GPT3 temperature and pressure predictions can result in unstable IWV retrievals based on the physical model described in Section 2.3, leading to significant errors. By using the LSTM time series model, it can effectively correct the stability of the sequence to address this issue, as shown in Fig. 5.18-5.21 (CMB denotes the combined model proposed in this research in each figure).

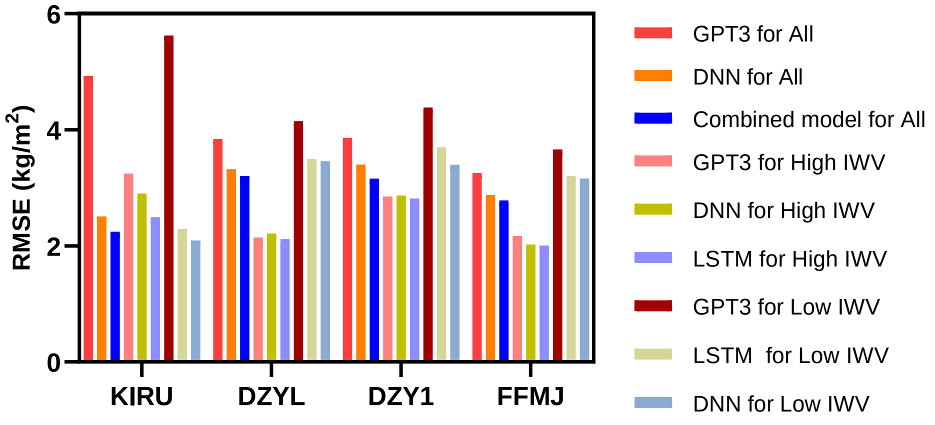


Fig. 5.17 RMSE comparison at different GNSS stations.

Table 5.5 RMSE, MB, MAPE and R^2 at different GNSS stations. Note that CMB denotes the combined model proposed in this section.

Metrics	**KIRU**	**DZYL**	**DZY1**	**FFMJ**
RMSE(GPT3) (kg/m^2)	4.93	3.84	3.86	3.25
RMSE(CMB) (kg/m^2)	2.24	3.20	3.15	2.78
MB(GPT3) (kg/m^2)	-2.00	-0.21	-0.20	-0.05
MB(CMB) (kg/m^2)	0.27	-0.07	-0.11	-0.16
MAPE(GPT3)	62.29%	25.20%	25.63%	21.06%
MAPE(CMB)	32.05%	20.57%	20.54%	17.93%
R^2(GPT3)	0.69	0.76	0.78	0.84
R^2(CMB)	0.85	0.84	0.84	0.88

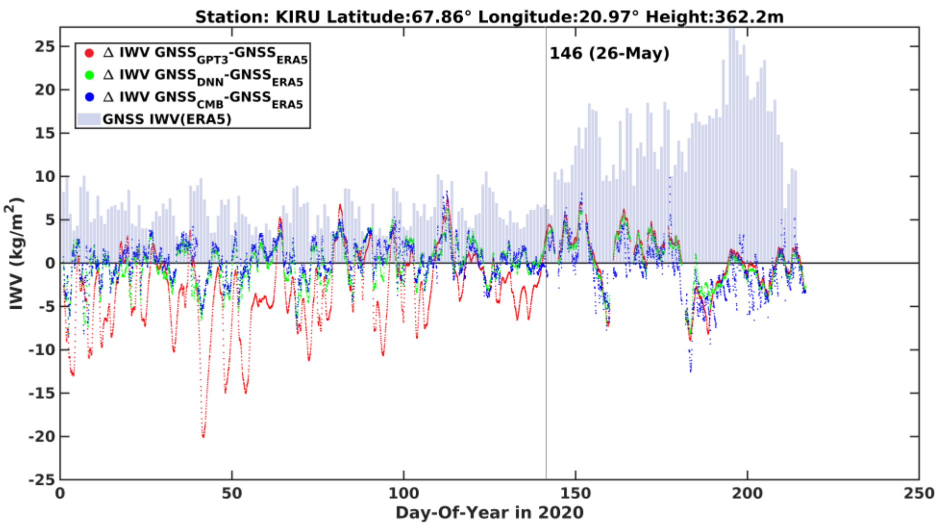


Fig. 5.18 IWV prediction difference comparison of KIRU, 2020. Before 26-May is the Low IWV period, and after that is the High IWV period.

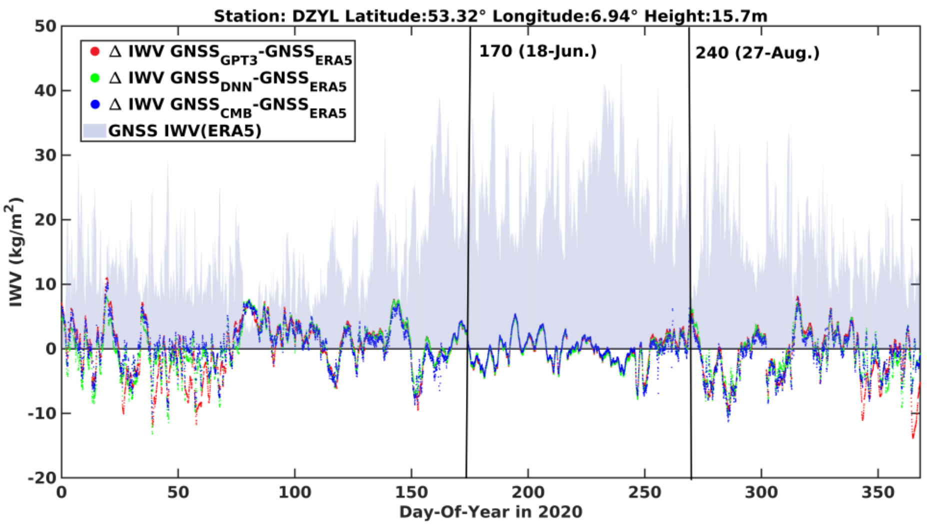


Fig. 5.19 IWV prediction difference comparison of DZYL, 2020. From 18-Jun. to 27-Aug. is the High IWV period and other days belong to the Low IWV period.

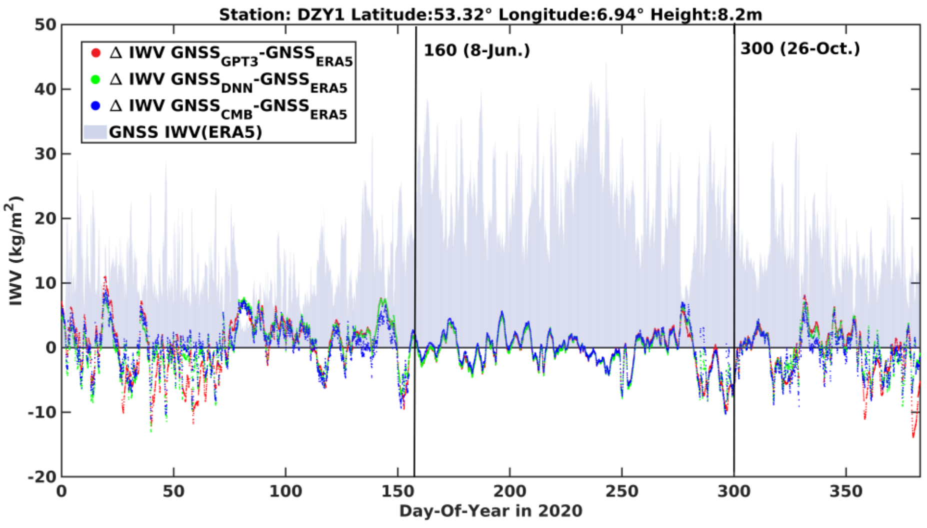


Fig. 5.20 IWV prediction difference comparison of DZY1, 2020. From 8-Jun. to 26-Oct. is the High IWV period, and other days belong to the Low IWV period.

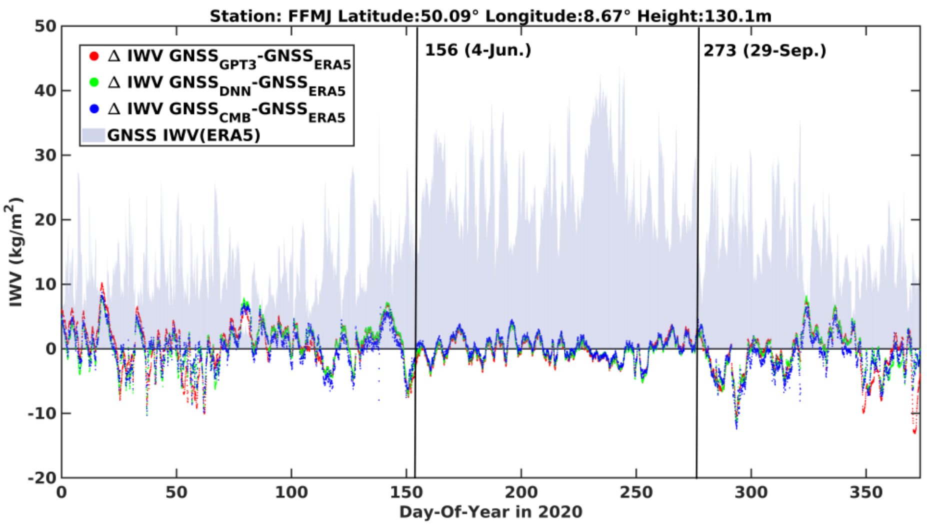


Fig. 5.21 IWV prediction difference comparison of FFMJ, 2020. From 4-Jun. to 9-Sep. is the High IWV period, and other days belong to the Low IWV period.

From Fig. 5.18-5.21, it can be observed that in Low IWV periods, the GPT3 model exhibits significant random errors during this period (red dots), while the model proposed in this paper significantly corrects this issue (blue dots). When the IWV values are higher, all three models make similar predictions. It is noteworthy that the difference of DNN and LSTM between classification boundaries for high and low IWV is not overly significant, as illustrated in Fig 5.18-5.21, where the blue and red points near the black line are close. While discontinuities at the edges of high/low IWV cases exist, their impact on inversion results remains within acceptable bounds. This can be attributed to in the first step of LSTM (right boundary) the hidden state is initialized as 0. Following the completion of the first iteration, the model proceeds normally. Therefore, this discontinuity will only occur 5 hours after the first bound and will not exert a significant influence.

5.2.4. Conclusion

This study proposed a deep learning method jointly using LSTM and feedforward DNN to perform real-time ZTD-IWV retrieval without actual meteorological observations; only GPT3 values are needed. By evaluating the proposed method at four different GNSS stations in the mid-to-high-latitude regions of Europe, it obtained RMSE values of 2.24, 3.20, 3.15, and 2.78 kg/m^2, respectively. Compared to the GNSS IWV retrieved from GPT3 with RMSE values of nearly 4 kg/m^2, the proposed method improves the real-time retrieval results at those stations.

Since climate patterns change slightly every year, it has been recommended to use at least three years of data from the year to be investigated for training the model to maintain stability. Increasing the training data period may be beneficial, depending on the similarity of the year to be investigated to its history. Considering that the stratospheric wind changes approximately every 2.5 years (Quasi-biennial oscillation) and the El Niño phenomenon occurs approximately every 3-5 years, therefore too long or too old data for training is not recommended.

6. Deep learning methods for Distributed Scatterers prediction

As described in Chapter 3, MT-InSAR can measure time series of surface deformation with millimeter-level accuracy. The joint processing of DS and PS based on the concept of SqueeSAR greatly alleviates the dependence of traditional PSI on high-quality surface scatterers. The preprocessed DS can be regarded as PS and used to monitor surface deformation in large-scale natural environments such as volcanoes, river valleys, and wilderness areas. However, a major bottleneck of the algorithm is the identification of DS, which requires homogeneity estimation for each pixel in the SAR image stack. Although DS can significantly increase the coverage of valuable information based on the physical properties of the Earth's surface, it requires a lot of computational effort, especially when modern high-resolution SAR sensors (such as Sentinel-1) generate increasingly large data stacks.

To address this challenge, this chapter introduces a deep learning method called DSPN that predicts whether a pixel qualifies for DSC before preprocessing. This method significantly improves processing efficiency. The predictions of DSC do not need to be completely accurate; they only need to avoid discarding true DS pixels while removing enough non-DS pixels. Since DS pixels naturally occur in groups and pixels are usually thinned out after merging PS and DS to reduce the amount of computation in subsequent processing steps, a slight prediction loss for DS pixels can be tolerated without compromising spatial coverage. This chapter demonstrates the performance of DSPN on six different topographic scenarios in North Rhine-Westphalia and Sicily, showing that the method effectively reduces computational requirements and ensures reliable prediction results.

The introduction of DSPN represents the first implementation of pre-classification for distributed scatterers before DS preprocessing. This approach pioneers the integration of deep learning technology into the DS preprocessing workflow to address the high computational burden and long processing times traditionally associated with DS identification. The primary content of this chapter is derived from a paper published by the author during his doctoral study:

Wang, D., Even, M., Kutterer, H. (2022). *Deep learning based distributed scatterers acceleration approach: Distributed scatterers prediction Net.* International Journal of Applied Earth Observation and Geoinformation, 115, 103112.

6.1. Problem statement and motivation

The preprocessing of DS, which involves grouping statistically homogeneous pixels, estimating coherence matrices, and extracting optimized phase histories, has been a focal point of InSAR research for over a decade (Even & Schulz, 2018). In essence, the goal is to convert DS into

PS-like pixels, which allows them to be used in any PS algorithm alongside actual PS. However, this process significantly increases computational demands. Consequently, methods to accelerate processing are highly sought after. On the technical side, advancements such as more powerful computers, parallelization, and optimized implementations can help meet this challenge. On the algorithmic side, grouping methods—such as the likelihood ratio test (Jiang & Guarnieri, 2020) or FSHP (Jiang et al., 2015)—have shown better performance in terms of speed and accuracy compared to the KS two-sample test. Additionally, the Eigendecomposition-based Maximum-likelihood-estimator of the Interferometric phase (Ansari et al., 2018) can quickly estimate DS signals while maintaining high-quality results. Nevertheless, estimating the phase history for each pixel in the SAR stack still demands significant memory to store intermediate calculations, and computational efficiency remains constrained by disk I/O, especially when the SAR stack cannot be fully loaded into memory and must be read in batches. Therefore, an acceleration scheme that considers computer implementation is urgently needed, which should avoid pixel-by-pixel calculations of the entire SAR images stack as much as possible.

This work is inspired by two key studies. The first is the Sequential Estimator proposed by Ansari et al. (2018), which demonstrated that the coherence matrix of DS can be iteratively estimated from a single SAR image, leveraging historical information without requiring the entire image stack. The second is research in land cover classification (LCC), which shows that certain land cover types—such as dense forests or water bodies, which are unlikely to serve as DS for SAR with shorter wavelengths like C-band or X-band—can be classified by polarized SAR characteristics. LCC by SAR and polarized SAR is a well-established field with promising results (Zhu et al., 2021), making it reasonable to use a few SAR images to identify non-DS pixels based on land cover characteristics.

Inspired by the above work, the goal of this study is to predict possible DSC before DS preprocessing using features extracted from a small number of SAR images and significantly reduce the computational burden by processing only the predicted DSC. In order to extract spatially homogeneous features with invariant properties, the CNN introduced in Section 4.4 can be utilized. The inherent properties of the convolution operation enable the network to focus on spatially invariant features, making the trained model suitable for reasoning across a variety of similar terrains. These features can be extracted from polarimetric SAR images with the worst coherence (strong vegetation) and best coherence (weak vegetation) at the acquisition time, as this represents the extreme cases of surface scatterer behavior and can provide enough information to distinguish DSC. These features can be complex and difficult to express in an explicit mathematical form. However, with the help of CNN, they can be implicitly extracted and used to distinguish DSC. The CNN is trained using supervised learning, as described in Chapter 4, with some DS preprocessed "data examples" for supervisory information. The input to the network consists of data features derived from the worst and best coherence polarimetric SAR images. It is worth noting that while LCC informed some aspects of this work, the main goal here is to classify DSC and non-DSC directly without relying on LCC as an intermediate step. Only certain input layers are selected based on insights related to LCC. The next section provides more details on the input features used and the supervision information required for the training process.

6.2. Methodology

6.2.1. Formal framework of DSPN

The framework of DSPN consists of three parts, namely "training set preparation", "training" and "mask prediction", as shown in Fig. 6.1. In the "training set preparation" section, the detail of the training set will be introduced. The focus of this section is how to compose the input features from the polarimetric SAR image and how to generate coarse labels from the preprocessed DS pixels as the ground truth. Then, DSPN is iteratively trained according to step 2 shown in Fig. 6.1; this section provides the architecture, training strategy and loss function of DSPN. After training, the model can be used to predict the mask of DSC, as described in the "mask prediction" section. The method of converting the output of DSPN into a mask that can be used by StaMPS is called coarse label correction, which is the focus of this section.

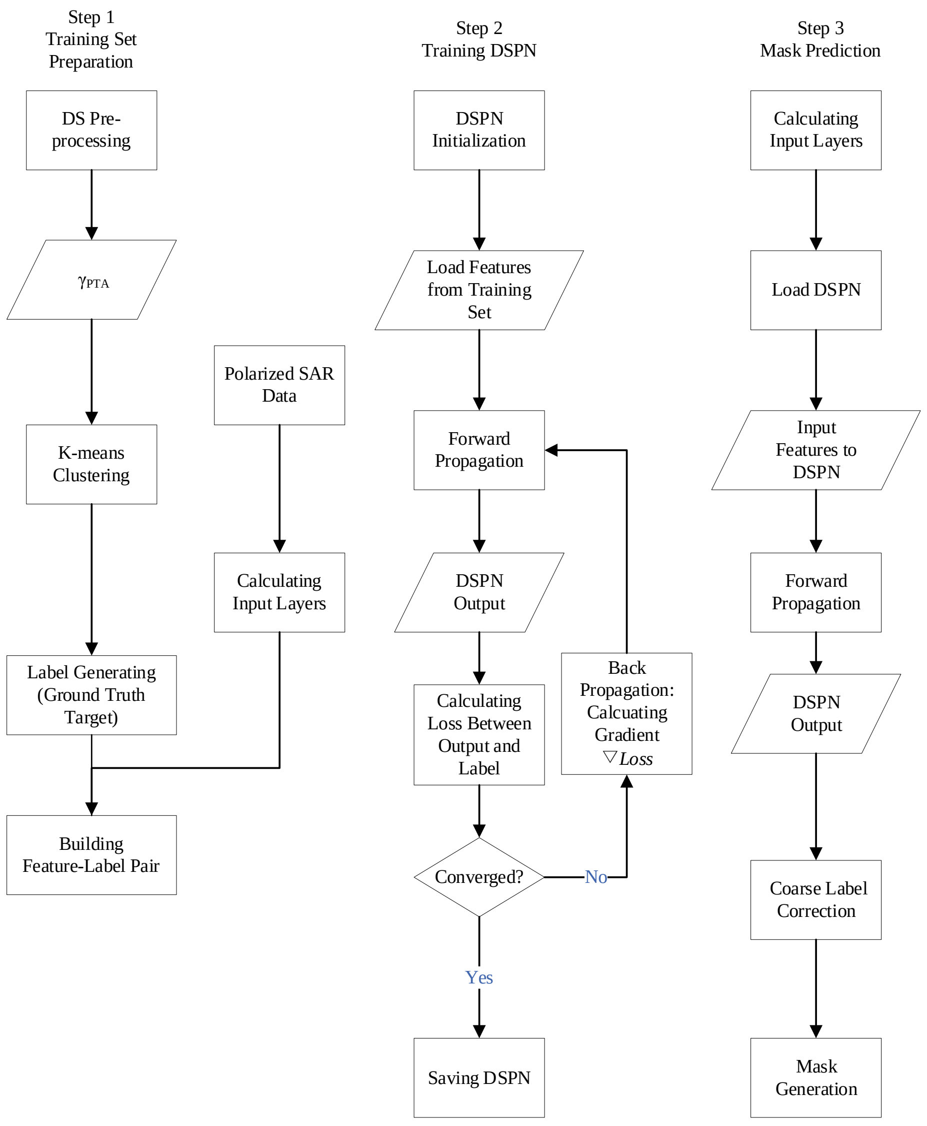


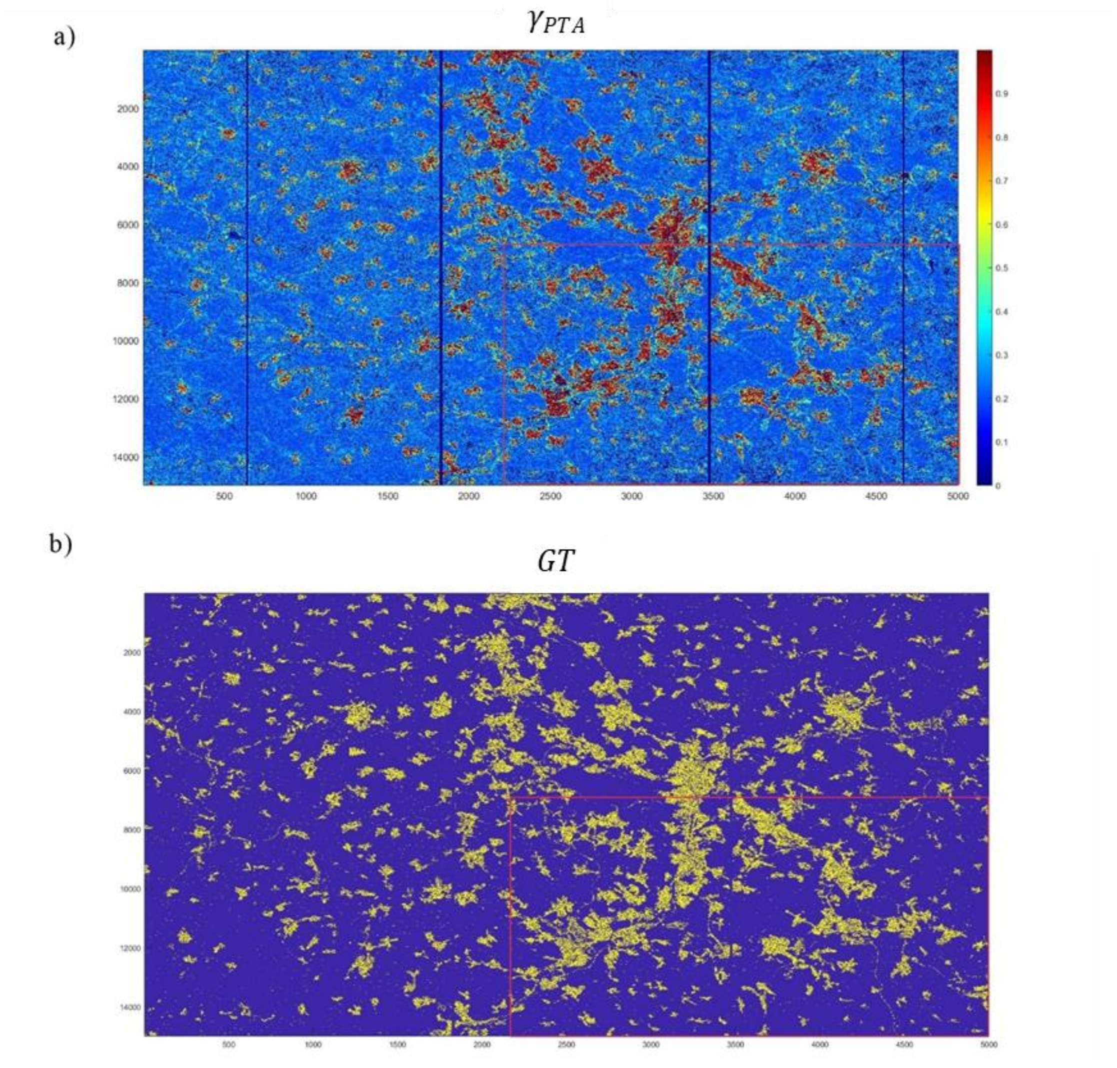
Fig. 6.1 The general flow chart of DSPN approach

6.2.2. Training Set Preparation

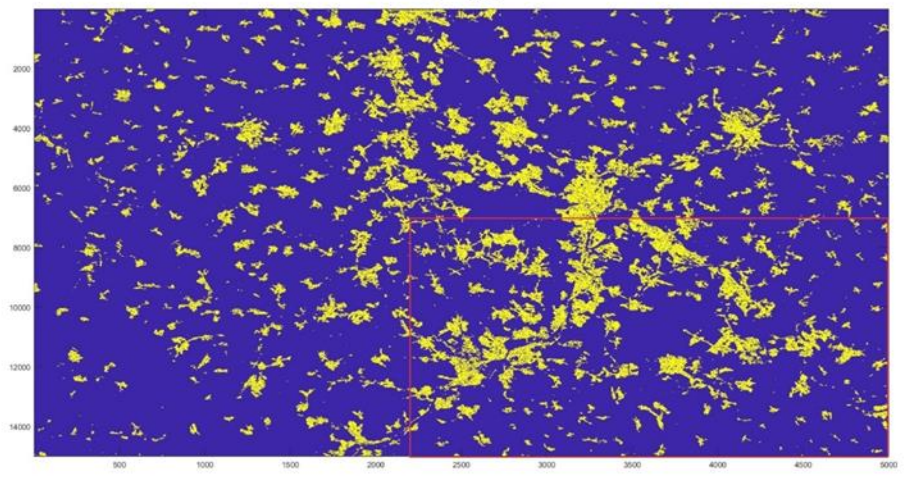
6.2.2.1. Ground Truth definition

How to supervise the training of neural networks is a central challenge in deep learning for geodesy. For DSC identification, it may seem straightforward to use the quality number γ_{PTA} as the ground truth. However, this would transform the neural network's output into a numerical regression task. As discussed in Section 3.4, γ_{PTA} is calculated from the historical optimal phase across the entire SAR images stack, meaning its precise value depends on the cumulative contribution of all images in the stack. Attempting to regress γ_{PTA} from only two images' results in substantial loss of information, leading to regression failure. In traditional DS preprocessing methods, DSC identification is often determined by a hard threshold, such as

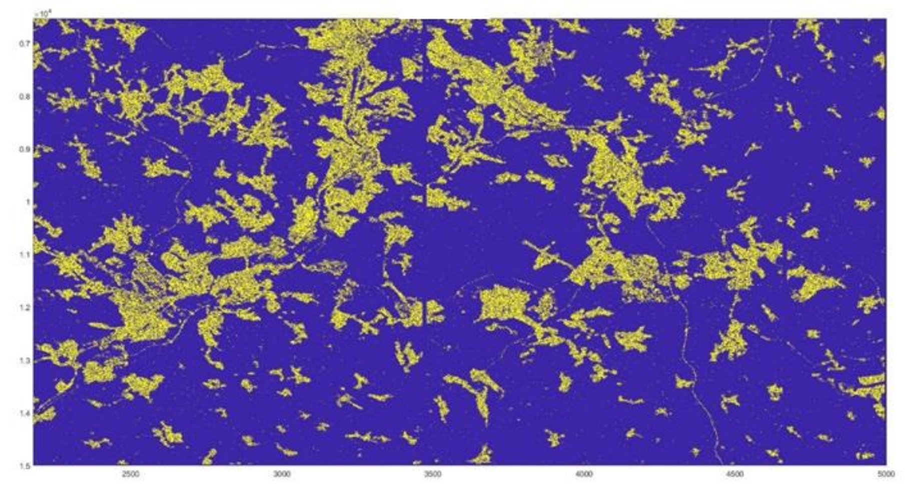
$\gamma_{PTA} > 0.4$. This suggests that the exact value of γ_{PTA} is less critical for DSC identification; instead, whether γ_{PTA} exceeds the threshold is the primary factor. Therefore, it is more reasonable to use a binary classification label based on this threshold rather than attempting to regress the exact γ_{PTA} value. However, the experiments of this study have shown that using γ_{PTA} as the binary ground truth label with general CNN, such as U-Net described in section 4.4, will cause the network to fail to capture important details with low γ_{PTA}, such as roads or highways, as shown in Fig. 6.2.



c) Pre_{U-Net}



d) GT



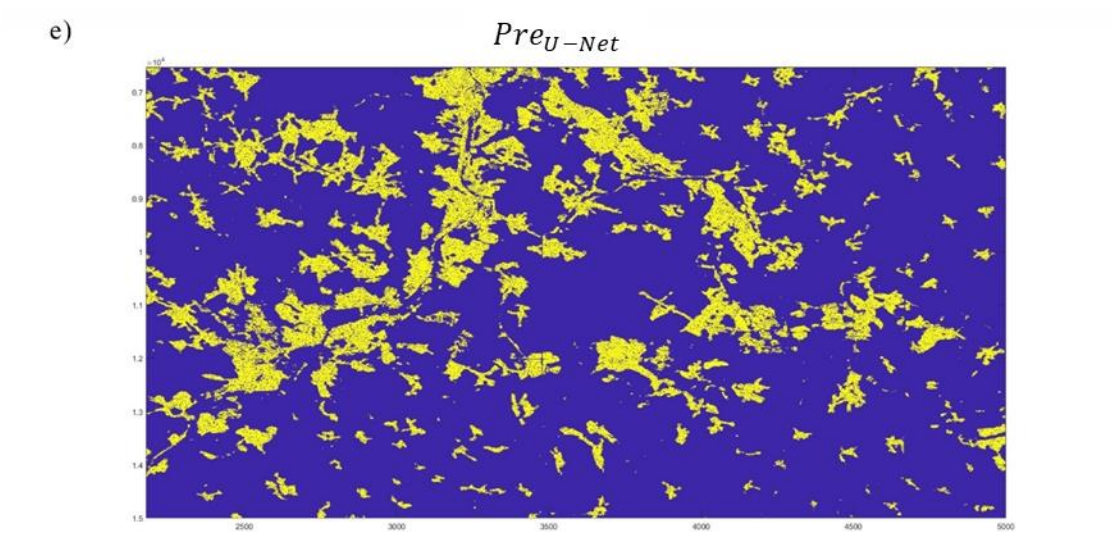


Fig. 6.2 An example of DS prediction by using U-Net (Ronneberger et al., 2015) trained by binary classification label as the ground truth. The red rectangle marks the detail region of the motorway, which has relevant low γ_{PTA}. a) The γ_{PTA} value in Saarland, calculated through the SqueeSAR approach. b) The binary classification label generated by $\gamma_{PTA} > 0.4$. c) The prediction result by U-Net. d) The details of binary classification label in red rectangle marked region. e) The details of prediction result by U-Net in red rectangle marked region.

After analyzing the distribution of γ_{PTA} in four different regions (Ruhr, Saarland, Ibbenbüren, and Sicily), an assumption is worth considering: It is supposed that the patterns of DSC follow a similar distribution for different scenarios. For instance, DSC can be broadly categorized into weak DS (e.g., wilderness, roads, highways, etc.) and strong DS (e.g., urban areas) according to their γ_{PTA} values. Although the exact γ_{PTA} values for the same type of DSC vary between different SAR stacks and parameter settings, their quantile values are close to each other. Based on this assumption, one approach to supervising the neural network with γ_{PTA} is to have the network learn from pseudo-categories of DSC, rather than regressing the exact value of γ_{PTA}. The pseudo-categories are derived from the γ_{PTA} by the following strategy:

1. As established in previous research, pixels with γ_{PTA} below a threshold (set here at 0.4) have a very low probability of being DSC (Even & Schulz, 2018). These pixels are therefore labeled as non-DS (0 for the mask) for the ground truth.

2. For pixels with $\gamma_{PTA} \geq 0.4$, the K-means clustering algorithm is employed to generate a multi-class coarse label mask for the ground truth. The coarse labels generated by K-means for those $\gamma_{PTA} \geq 0.4$ pixels can be seen as the pseudo-categories of different types of DSC. This enables the network to learn a more defined decision hyperplane, improving its ability to classify various DSC types and increasing its sensitivity to weak DSC pixels.

Although the hard threshold of 0.4 might be considered somehow conservative, it is suitable in this context since retaining more DSC pixels is less harmful than losing coverage, because those false positive pixels will be discarded in the following step of StaMPS. If using a more strict threshold like $\gamma_{PTA} \geq 0.7$ (Samiei-Esfahany, 2017), this will cause the network to be too strict in identifying DS, and most DSC will be rejected, causing the available DS to be too sparse. Since K-means is a non-convex algorithm, determining the optimal number of categories, K, is done through a systematic "trial and error" approach. Experiments suggest that if the number of clusters exceeds a maximum threshold K_{max} (with $K_{max} \leq 6$ being recommended), convergence becomes difficult. Therefore, the optimal K is determined by testing values between 2 and K_{max}. To determine the optimal K, the Calinski-Harabasz index (Caliñski & Harabasz, 1974), Silhouette analysis (Rousseeuw, 1987), and the Elbow curve method (Ketchen & Shook, 1996) can be used to generate the potential optimal K. If the potential optimal K generated by those three methods is different, the value whose histogram best fits the distribution of γ_{PTA} is chosen. Additionally, if a cluster is significantly smaller than others, it is merged with the nearest larger cluster, ensuring the neural network can effectively learn the features of the various classes. The mask generated by the K-means clustering algorithm reflects the scattering behavior of surface scatterers in the given scene, though they do not correspond to specific ground object types—hence the term "pseudo-class." This distinguishes the task from LCC. For DSC prediction, it is not necessary to determine the exact source of the DSC (e.g., whether it originates from a wilderness or road surface). The purpose of the clustering labels is to distinguish DSC with different scattering behaviors, allowing the network to capture finer details, especially for weak DSC pixels.

Following the above-mentioned processing, a multi-label mask corresponding to non-DSC and various types of DSC is generated. This mask serves as the ground truth for neural network training. Consequently, the DSC prediction task is reformulated as a pixel-wise classification problem within the framework of semantic segmentation.

6.2.2.2. Input features definition

To enable end-to-end learning of DS patterns based on the statistical characteristics of raw signals, DSPN aims to extract features directly from SAR images. Unlike natural RGB images, however, in the raw SAR images, only the amplitude information makes sense, making it difficult to derive physically meaningful insights from unprocessed radar amplitudes. Fortunately, studies in SAR for vegetation (Nikaein et al., 2021) and crop (Mestre-Quereda et al., 2020) classification have demonstrated that coherence and backscatter are complementary and that dual-polarization SAR data yields superior results compared to single-polarization. This insight inspired DSPN to utilize polarimetric SAR images for extracting DS features. Besides, findings from LCC indicate that dual-polarization data enhances road extraction performance (Xiao et al., 2019). Roads, though often of lower quality in DSC, are particularly valuable and can contribute to additional coverage, e.g., in Central Europe (Zhang et al., 2019). Furthermore, in vegetated areas, the coherence of DS is significantly influenced by seasonal variations, particularly for short-wavelength radars such as C-band or X-band, due to the strong interaction of radar signals with the vegetation canopy. During periods of dense foliage, the radar signal undergoes extensive volume scattering within the canopy, whereas during periods of leaf fall, the signal can penetrate the canopy more effectively and produce double-bounce reflections from the ground. The objective of DSPN is to predict DSC using a minimal number of SAR images. To achieve this, it is essential to account for coherence information under conditions of both high coherence (sparse canopy, short temporal baseline) and low coherence (dense canopy, long temporal baseline).

Inspired by the above work and experimental experience, DSPN finally selected the following 9 features as its input: speckle filtered intensities of the master regarding VV and VH; real and imaginary parts of coherences from an acquisition of a month with low coherence and a month with high coherence; median amplitude over the stack; filtered amplitude dispersion image of the stack; and coefficient of variation of median amplitude. Among them, the intensities are the output of the polarimetric speckle filtering function of the Sentinel 1 toolbox (Improved Lee Sigma filter using default parameters one look, window size 7×7, sigma 0.9, target window size 3×3). The coherences are calculated for each acquisition from a month with low coherence (dense canopy, long temporal baseline) and a month with high coherence (sparse canopy, short temporal baseline). For the calculation of the coherences, an 11×3 window is used. The median amplitude and the amplitude dispersion image of the stack do not pose an additional computational burden as they are used as the background image for plotting or are required for PS selection and are calculated either way. The median amplitude gives a low-noise version of backscatter. Low amplitude dispersion is often used as an indicator of phase stability that helps detect PSC. In order to obtain a layer with easier-to-classify information a median filter (window size 5×3) that exempts pixels with amplitude dispersion smaller than 0.4 is applied. The size of the median filter is a trade-off between losing resolution and being large enough to observe the desired smoothing effect. The coefficient of variation of median amplitude (7×3 window) is calculated to highlight in particular roads.

6.2.3. Training DSPN

6.2.3.1. Net Architecture

Given that the network must identify the class of each pixel, the use of an encoder-decoder network architecture is an intuitive choice. The DSPN structure comprises two main components: the encoder branch and the decoder branch. For the encoder, a convolutional network backbone with multi-level feature extractors, such as AlexNet (Krizhevsky et al., 2012), GoogLeNet (Szegedy et al., 2015), or VGG (Simonyan & Zisserman, 2015), can be considered. However, it is more recommended to use ResNet (He et al., 2016) as the backbone due to its ability to extract features across four different scales—ranging from global to local—via its four ResBlocks (implemented using Bottlenecks (He et al., 2016)), effectively addressing the vanishing gradient problem.

The decoder path consists of multiple decoding units, each containing a set of 3×3 convolutions and an up-sampling unit. To avoid the "checkerboard artifacts" (Odena et al., 2017) commonly caused by deconvolution, the deconvolution is replaced with bilinear interpolation and a 1×1 convolutional layer to construct the up-sampling units. Unlike V-net (Milletari et al., 2016) and U-net (Ronneberger et al., 2015), dense concatenation (Zhou et al., 2018) is used to merge features from different decoding units in various decoding paths rather than directly concatenating them. This approach leverages the fact that features from different decoding paths have varying receptive fields, allowing dense skipping to more effectively integrate encoding information at multiple scales. This enhances the network's ability to capture finer details, such as small roads or motorways. The overall architecture of DSPN is illustrated in Fig 6.3, with detailed layer specifications provided in Table 6.1.

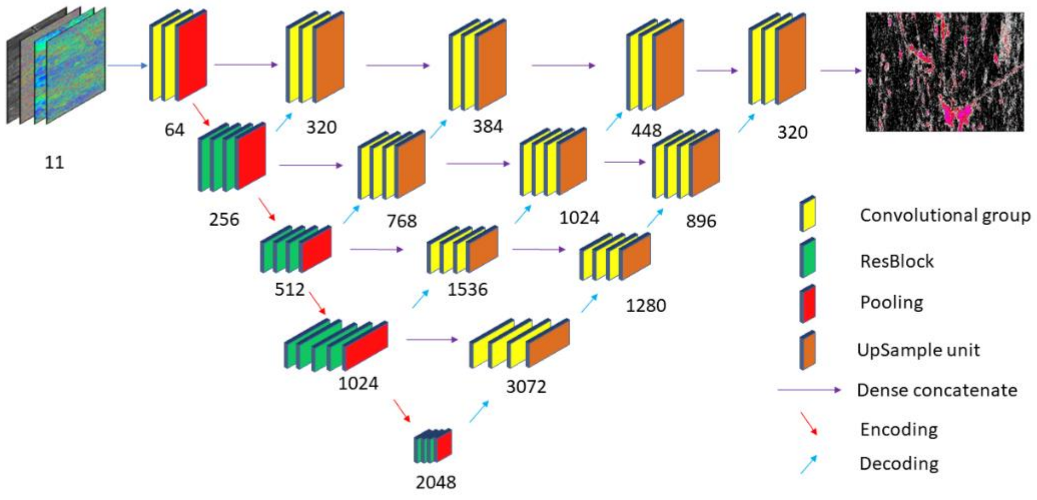


Fig. 6.3 DSPN architecture. The number below each block indicates the input channel for that block.

Table 6.1 Architecture details of DSPN

	Encoder	Decoder
Layer 1 (11)	Encoder 1: Conv (11->64)	Decoder 1: Conv (UP(Decoder2.4(32))->K)
Layer 2 (64)	Encoder 2: Bottleneck (64->256)	Decoder 2.1: Conv (UP(Encoder2(256))+Encoder1(64)->64) Decoder 2.2: Conv(Encoder1(64)+Decoder2.1(64)+UP(Decoder3.1(256))->64) Decoder 2.3: Conv(Encoder1(64)+Decoder2.1(64)+Decoder2.2(64)+UP(Decoder3.2(256))->64) Decoder 2.4: Conv(Encoder1(64)+Decoder2.1(64)+Decoder2.2(64)+Decoder2.3(64)+UP(Decoder3.3(64))->32)
Layer 3 (256)	Encoder 3: Bottleneck (256->512)	Decoder 3.1: Conv(UP(Encoder3(512))+Encoder2(256)->256) Decoder 3.2: Conv(Encoder2(256)+Decoder3.1(256)+UP(Decoder4.1(512))->256) Decoder 3.3: Conv(Encoder2(256)+Decoder3.1(256)+Decoder3.2(256)+UP(Decoder4.2(128))->64)
Layer 4 (512)	Encoder 4: Bottleneck (512->1024)	Decoder 4.1: Conv(UP(Encoder4(1024))+Encoder3(512)->512) Decoder 4.2: Conv(Encoder3(512)+Decoder4.1(512)+UP(Decoder5(256))->128)
Layer 5 (1024)	Encoder 5: Bottleneck (1024->2048)	Decoder 5: Conv(UP(Encoder5(2048))+Encoder4(1024)->256)

The notation OPERATOR(X->Y) represents data propagation, where X denotes the input of this operator and Y denote the output. The notation Conv represents the convolutional layer, and the notation UP represents the up-sample layer. For example, the input channel of Decoder 4.2 is 1280, which consists of the output of Encoder 3 (512 channels), the output of Decoder 4.1 (512 channels), and the up-sampled output of Decoder 5 (256 channels). After a 1×1 convolutional layer, the output of Decoder 4.2 has 128 channels.

6.2.3.2. Loss function

With the coarse label generation approach mentioned above, the DSC prediction task is reframed as a semantic segmentation problem, making the cross-entropy loss function a natural choice. However, this standard loss function does not account for the varying contributions of different misclassified labels, particularly in cases where DSC is incorrectly categorized into a pseudo-class that is also part of DSC but differs from the Ground Truth, which does not impact on the overall DSC classification outcome. Given that the primary goal of DSPN is to determine whether a pixel belongs to DSC, this study proposes using a weighted cross-entropy loss function $Loss_{WCE}$ for coarse label learning, as described below:

$$Loss_{WCE} = \underset{W}{\text{argmin}} H_{WCE}(P||Q) = -\underset{W}{\text{argmin}} \frac{1}{N} \sum_{n=1}^{N} w_k P(gt_n) \log Q(y_n) \tag{6.1}$$

$$w_k = \begin{cases} 1 & if\ k = 0\ (non-DSC) \\ \frac{N_k}{\sum_{k=2}^{K} N_k} & else \end{cases} \tag{6.2}$$

where $P(gt_n)$ is the Multinoulli distribution of pixel x_n belonging to DSC which generated by coarse label mentioned above, and $Q(y_n)$ refers to the probability distribution transferred by SoftMax function (Eq. 4.11) of DSPN output y_n, N_k is the number of points belonging to category k and class $k = 0$ denotes the non-DSC.

6.2.3.3. Training strategy and optimizer

Once the DSPN structure is constructed, the network weights need to be initialized. The Kaiming initialization method (He et al., 2015) is recommended for this purpose. After initialization, the network can be iteratively trained until convergence using the back-propagation algorithm detailed in Section 4.3. For optimizers, it is advisable to use either Adam or SGD, which are also described in Section 4.3. While Adam is known for its ability to achieve fast convergence by employing an adaptive learning rate, it has been reported that its generalization ability is not superior to that of SGD (Keskar & Socher, 2017). Therefore, it is suggested to use the Adam optimizer during the initial stages of training for a warm-up phase, followed by momentum-based SGD for fine-tuning to achieve improved accuracy and generalization performance. If a pre-trained model is available, SGD can be directly applied for transfer learning, eliminating the need to reinitialize and retrain the network when processing new datasets. Transfer learning offers faster convergence compared to retraining from scratch, thereby enhancing DSPN's generalization capabilities while reducing training time. When the network converges, its weights can be saved for mask prediction.

6.2.4. Mask Prediction

Once the network has been trained and its weights are obtained, DSPN can use these weights to make DSC predictions for any location within a similar scene. After loading the saved network weights, the input features of the image to be predicted are fed into DSPN, which performs forward propagation. The network then outputs a tensor O of dimensions

$Az \times Rg \times K$, where Az and Rg is the Azimuth and Range of the input image, and $O[a,r,k]$ following a SoftMax conversion represents the probability that DSPN predicts the pixel at position $[a,r]$ as pseudo-category k. As the primary objective is to filter out non-DSC pixels rather than generate a coarse label classification map, the final mask can be derived from the SoftMax-converted tensor O using the following coarse label correction method:

1. Generate the binary mask by:

$$Mask[a,r] = \begin{cases} 0 & O[a,r,1] > \sum_{i=2}^{K} O[a,r,i] \\ 1 & otherwise \end{cases} \tag{6.3}$$

2. Morphological correction:

Since DSC consists of clusters of points exhibiting the same scattering mechanism, a true DS point is expected to have at least 20 neighborhoods (Ferretti et al., 2011), this implies that isolated points with small neighborhoods are unlikely to be DS. To maintain adequate coverage, the neighborhood threshold for the DSPN mask is set to 5, meaning that points with fewer than 5 neighbors will be classified as non-DS. Subsequently, the mask is dilated to prevent the loss of DSC near the edges.

After coarse label correction, the generated mask can be used as an indicator for DSC. Then, the processing efficiency can be significantly improved by preprocessing only the DSC covered by the mask.

6.3. Experiment

6.3.1. Data & Experiment environment

To validate the proposed method, two Sentinel-1 IW image stacks from North Rhine-Westphalia, Germany, and Sicily, Italy, were utilized to assess the performance and generalization capability. The specific details of these image stacks are provided in Table 6.2. Six distinct test areas were selected from these two data stacks to evaluate DSPN's performance across different topographies. The details of these test areas are listed in Table 6.3. The image patches cropped for training were taken from the entire scene, excluding the test areas.

Table 6.2 SAR images stacks details for North Rhine-Westphalia and Sicily

Dataset	**Path**	**Master date**	**Period**	**Low coherence date**	**High coherence date**	**Size**
North Rhine-Westphalia	15 Ascending	2018.3.01	2018.1.06-2021.3.21	2017.5.11	2018.2.17	4000*9000
Sicily	44 Ascending	2017.1.07	2017.10.08-2018.9.29	2017.4.19	2017.1.13	15000*11750

To construct the training and test sets, DS preprocessing is performed with the following parameters or sub-algorithm selections: For grouping, a search window of size 21 pixels times 21 pixels was used. As a criterion of similarity, the generalized likelihood ratio test with a confidence level of 99% was applied (cp. (Jiang & Guarnieri, 2020) for some theory). The phase history was estimated with the help of phase triangulation coherence maximization (Ferretti et al., 2014) from the coherence matrix, which was obtained from the sample covariance matrix. In order to mitigate biases, the entries of the coherence matrix were taken to the power of two (Ferretti et al., 2014). Furthermore, sequential estimation with a mini stack size of 20 acquisitions was employed (Ansari et al., 2017). These choices were based on a study in two parts on how to tune DS pre-processing (Even, 2021, 2022).

The proposed method above was implemented on a deep learning PC with a single NVIDIA RTX 3090 GPU and Intel(R) Core i7-10700 CPU, 8 threads. The DS pre-processing and coarse label generation was implemented by using MATLAB R2020b. The DSPN, weighted coarse label cross-entropy, and coarse label correction were implemented by using Pytorch 1.7.0 build-in operators. Considering the GPU memory constraints, the input layers and corresponding ground truth labels were cropped into 500×500 pixel-sized patches subset stacks.

For North Rhine-Westphalia dataset, the model of DSPN was trained with 2000 epochs iteratively. The learning strategy is: Use of Adam optimizer with 0.001 initial learning rate, and for every 500 iterators, the learning rate will decay by multiplying 0.7. Training North Rhine-Westphalia model required approximately 49 hours of our environment. For each test region, the mask generating time was no higher than one minute. Although SAR images in Sicily are susceptible to complex terrain such as slope shadows, the generalization ability of DSPN enables the network to work well with only transfer learning. To evaluate the generalization ability of DSPN, we use the fine-tuning strategy, that is: using North Rhine-Westphalia model as initial weight and fine-tune training 500 epochs with SGD, 0.0005 initial learning rate and decay to 0.0001 after 250 epochs. Although the fine-tuning training nearly took 15 hours, again the mask was generated in one minute when the training finished.

Table 6.3 Test regions details for North Rhine-Westphalia and Sicily

Dataset	Test region	Topography	Size
North Rhine-Westphalia	Wickede (Ruhr)	River , Countryside	500×1000
North Rhine-Westphalia	Hamm	Urban	500×2000
North Rhine-Westphalia	Münster	Motorways	500×1000
Sicily	Arenella	Urban, Harbor, Beach, Sea	1500×500
Sicily	South side of Etna	Crater-volcano (slope and shadow)	3500×3000
Sicily	Rocche d'Argimusco	Forests (slope and shadow)	1500×1500

6.3.2. Results of coarse label generation

After searching, the best K for North Rhine-Westphalia is 3, and the best K for Sicily is 5, a higher value will cause the algorithm to fail to converge. The clustering result is shown in Fig. 6.4 and Fig. 6.5.

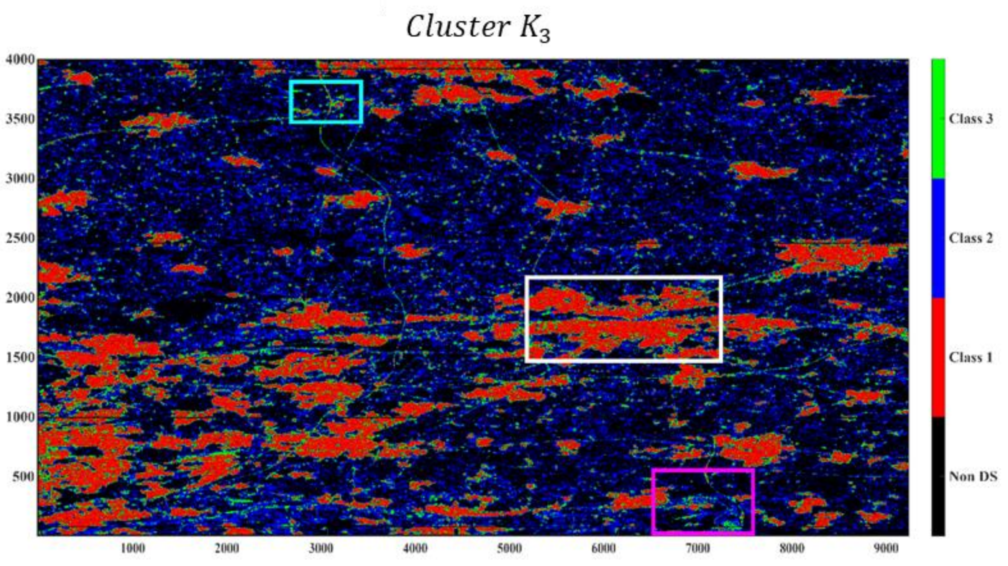


Fig. 6.4 The clustering result for North Rhine-Westphalia; the white rectangle marks the test region in city Hamm, the cyan rectangle marks the test region in the motorway intersection near Münster and the purple rectangle marks the test region in Wickede (Ruhr).

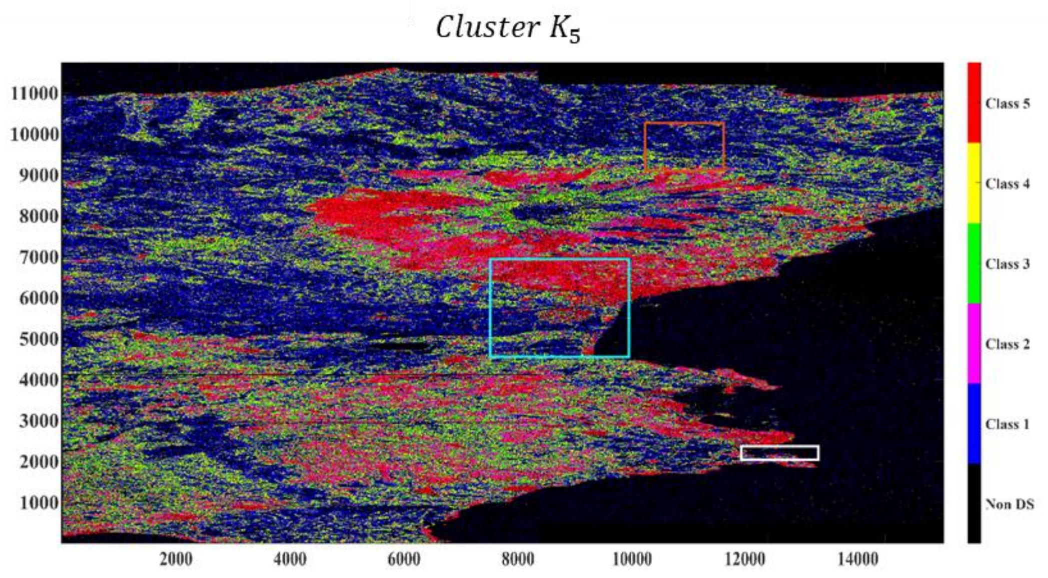


Fig. 6.5 The clustering result for Sicily; the white rectangle marks the test region in Arenella, the cyan rectangle marks the test region in the south side of Etna and the brown rectangle marks the test region in Riserva Naturale Orientata Bosco di Malabotta. Rocche d'Argimusco.

6.3.3. Evaluation of DSPN

To evaluate the performance, the numbers of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) samples are used to calculate Accuracy (Acc), Precision (Pre), Recall (Rec), Negative Predictive Value (NPV), and False Negative Rate (FNR) as the metrics of mask quality with following formulas:

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{6.4}$$

$$Pre = \frac{TP}{TP + FP} \tag{6.5}$$

$$Rec = \frac{TP}{TP + FN} \tag{6.6}$$

$$NPV = \frac{TN}{TN + FN} \tag{6.7}$$

$$FNR = \frac{FN}{FN + TP} \tag{6.8}$$

Since the objective is to create a mask that accelerates DS preprocessing, type II errors (misclassifying DSC as non-DSC, represented as false negatives) are more detrimental than type I errors (misclassifying non-DSC as DSC, represented as false positives). While type I errors only introduce unnecessary computation, type II errors result in the loss of DS points, which negatively impacts subsequent DS processing. The mask generated without coarse label correction may still contain some false positive pixels, but these typically appear as isolated noise points rather than clusters of scatterers with similar properties. After applying morphological postprocessing, the remaining false positive pixels are corrected during the later stages of DS preprocessing. To assess the impact of these errors, the False Negative Rate is used as a metric to evaluate the loss rate (type II error), reflecting the proportion of missing DSC pixels.

6.3.3.1. Performance of DSPN in North Rhine-Westphalia

The evaluation result of three test regions in North Rhine-Westphalia is shown in Table 6.4. The proposed DSPN achieved a better performance in the urban area (Hamm), which has 98.41% accuracy and only 2.09 % FNR. The detail of γ_{PTA} after DS preprocessing with or without our approach is shown in Fig. 6.6 to Fig. 6.8. Even though in the rural area or suburban highway (Wickede (Ruhr) and Münster) the accuracy of the proposed approach was lower than in the urban area, our approach still performs well in identifying important objects (airports, rivers, roads, etc.) as shown in Fig. 6.7b and Fig. 6.8b.

Table 6.4 Evaluation Results of North Rhine-Westphalia.

Region	**TP**	**TN**	**FP**	**FN**	**Acc (%)**	**Pre (%)**	**Rec (%)**	**NPV (%)**	**FNR (%)**
Wickede (Ruhr)	172830	295501	0	31669	93.67	100	84.51	90.32	15.49
Münster	181097	292371	0	26532	94.69	100	87.22	91.68	12.78
Hamm	743338	240795	0	15867	98.41	100	97.91	93.82	2.09

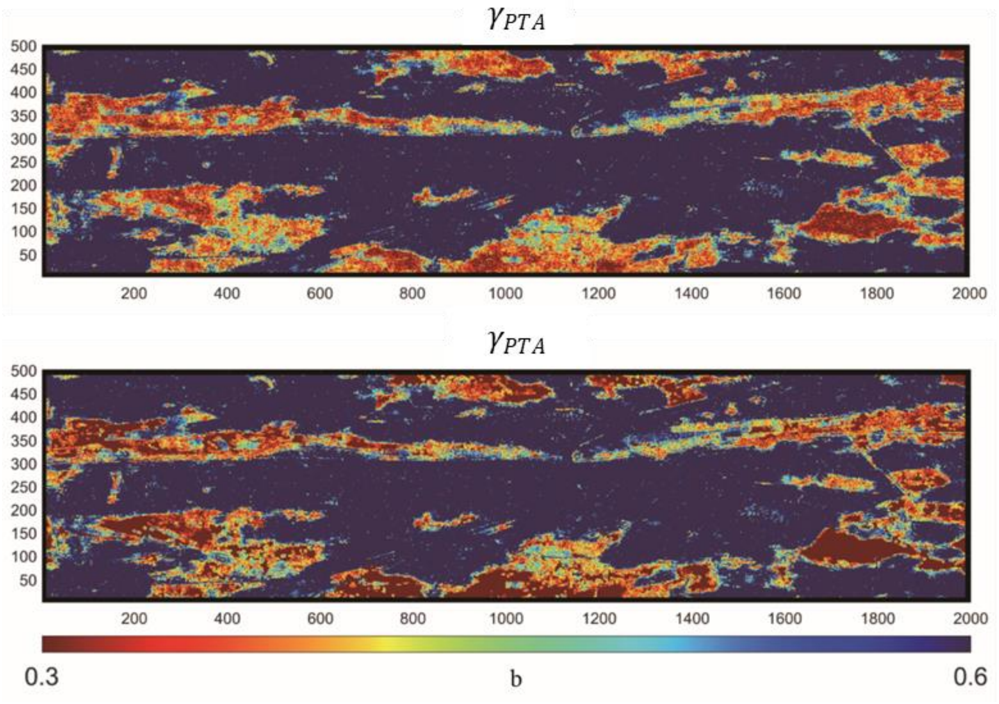


Fig. 6.6 γ_{PTA} after preprocessing in Hamm – (a) without DSPN; (b) with DSPN.

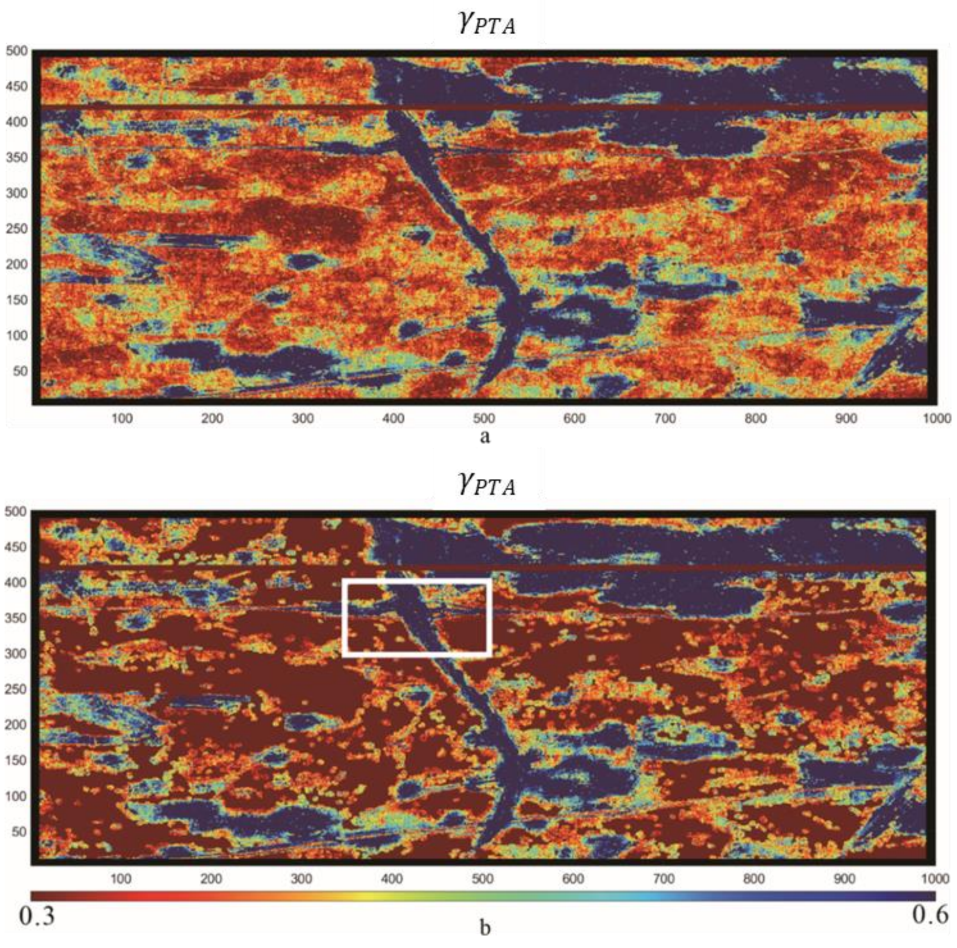


Fig. 6.7 γ_{PTA} after preprocessing in Münster – (a) without DSPN; (b) with DSPN; the white rectangle marks the overpass between Highway 1 and Highway 43.

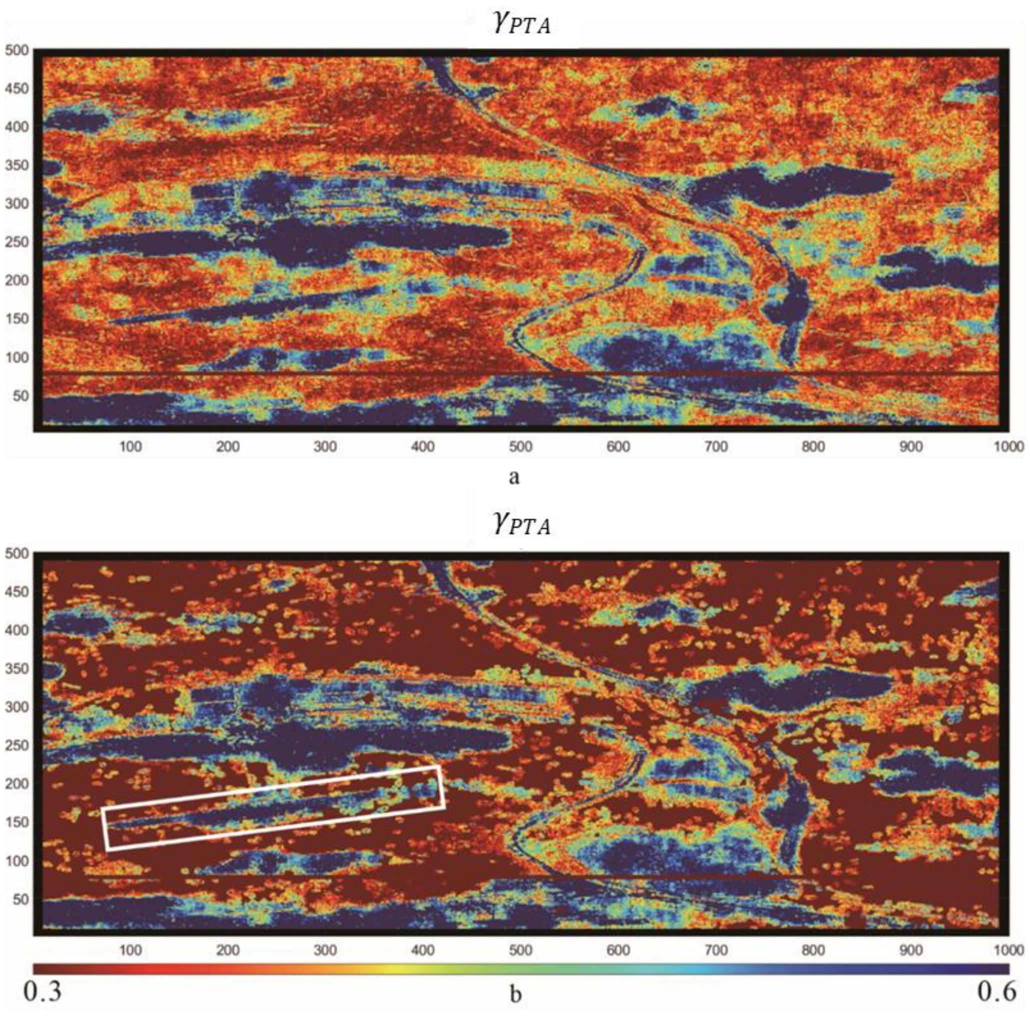


Fig. 6.8 γ_{PTA} after preprocessing in Wickede (Ruhr) – (a) without DSPN; (b) with DSPN; the white rectangle marks the airport Arnsberg Menden.

6.3.3.2. Performance of DSPN in Sicily

The evaluation results of the test regions in Sicily are shown in Table 6.5 and Fig. 6.9 to Fig. 6.11. After 500 epochs of fine-tuning training, DSPN also predicted well on the sea – a surface class not appearing in North Rhine-Westphalia (achieved 99.83% accuracy and 1.41% FNR in Arenella). For mountain areas with complex terrain, the mask can still retain most of the DSC points (achieved 98.40% accuracy and 2.23% FNR in the South side of Etna).

Table 6.5 Evaluation results of Sicily.

Region	TP	TN	FP	FN	Acc (%)	Pre (%)	Rec (%)	NPV (%)	FNR (%)
Arenella	86778	661978	0	1244	99.83	100	98.59	99.81	1.41
South side of Etna	7369239	2962704	0	168057	98.40	100	97.78	94.63	2.23
Rocche d'Argimusco	880722	1281276	0	88002	96.09	100	90.92	93.57	9.08

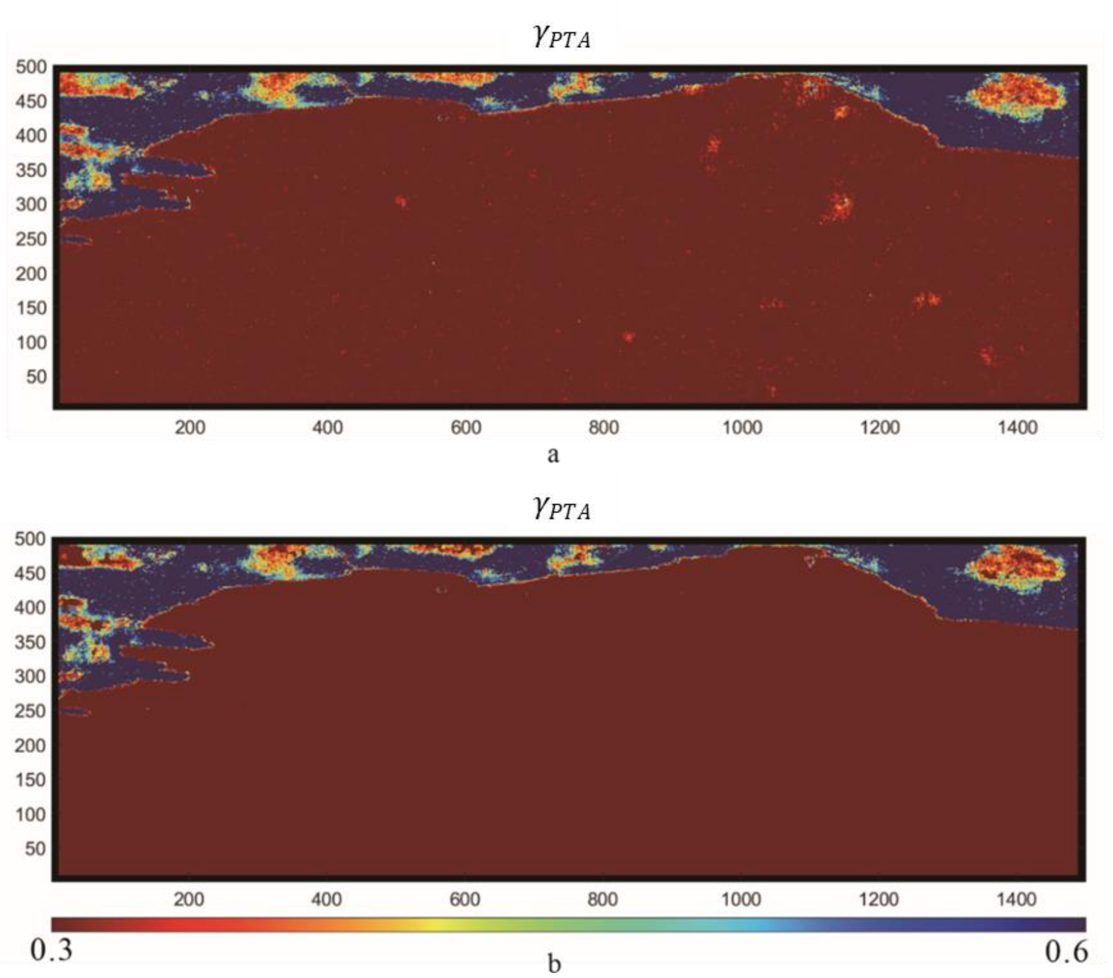


Fig. 6.9 γ_{PTA} after pre-processing in Arenella – (a) without DSPN; (b) with DSPN.

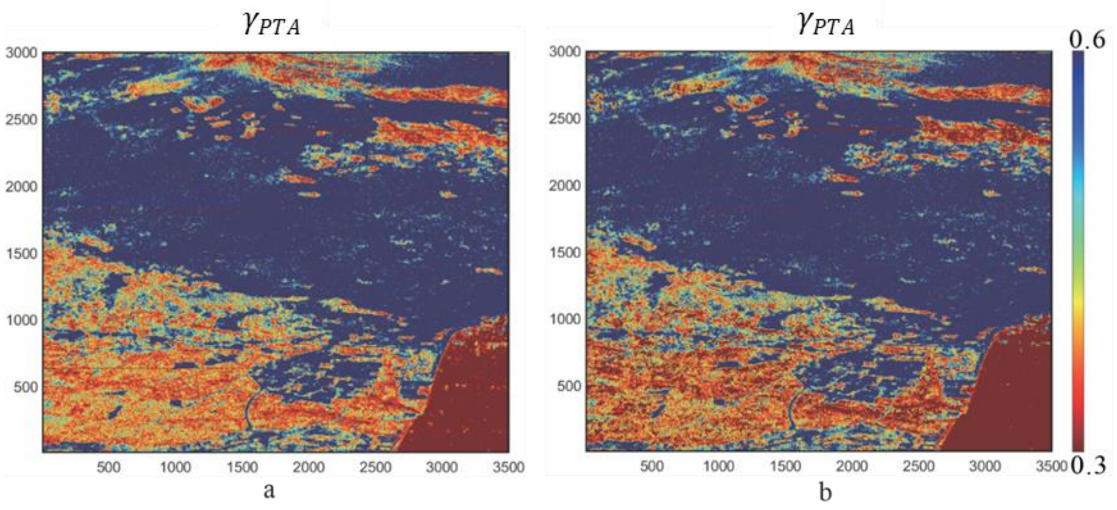


Fig. 6.10 γ_{PTA} after pre-processing in South side of Etna – (a) without DSPN; (b) with DSPN.

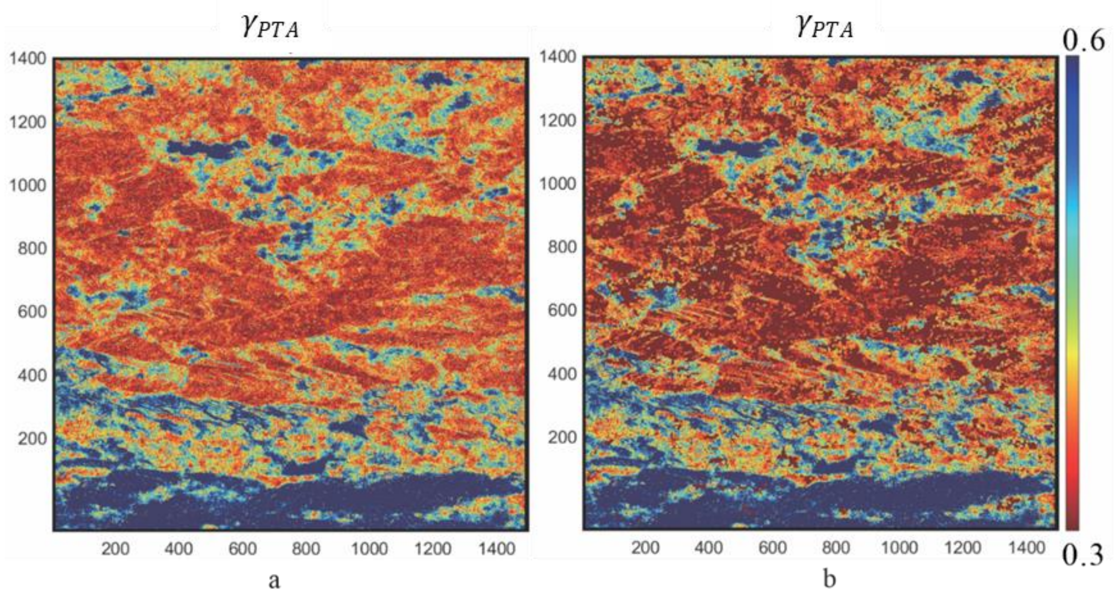


Fig. 6.11 γ_{PTA} after pre-processing in Riserva Naturale Orientata Bosco di Malabotta. Rocche d'Argimusco – (a) without DSPN; (b) with DSPN.

6.3.4. Time Costing of DSPN

Since the acceleration performance of DSPN varies across application scenarios, it is necessary to reasonably assess the efficiency of the speed improvement it offers. Given that DS preprocessing typically employs multi-threading technology for parallel computing, it is necessary to compare the algorithm's running time with both the actual running time T_a and the total computation time T_t (sum of all thread computation time, including scheduling costs) of DS pre-processing with or without using the proposed approach. The results of this comparison are presented in Table 6.6. From the analysis of the six test areas, it is evident that the higher the proportion of non-DSC in the scene, the greater the acceleration DSPN provides for preprocessing.

Table 6.6 The comparison of actual running time $\boldsymbol{T_a}$ and the total computation time $\boldsymbol{T_t}$ in test regions

Region	$\boldsymbol{T_a(with\ DSPN)}$ **/sec**	$\boldsymbol{T_a(without\ DSPN)}$ **/sec**	$\boldsymbol{T_t(with\ DSPN)}$ **/sec**	$\boldsymbol{T_t(without\ DSPN)}$ **/sec**
Wickede (Ruhr)	9861	14076	48089	90018
Münster	9633	14222	57783	88043
Hamm	19207	22405	117603	133245
Arenella	7663	13303	8358	66271
South side of Etna	80585	96596	1392960	1606706
Rocche d'Argimusco	40549	48349	218120	293939

6.4. Discussion

6.4.1. Acceleration efficiency

Since the actual running time T_a is influenced by bottlenecks in data loading and parallel computing scheduling, the proposed method can reduce the actual processing time by approximately 14.27% (in Hamm, with only 9.12% non-DSC) to 42.40% (in Arenella, with 83.63% non-DSC). Excluding the data loading factor, the main reason for the discrepancy between the actual acceleration and the percentage of non-DSC is that the MATLAB parallel computing mechanism does not distribute the computational load evenly across all threads. The actual running time T_a is determined by the slowest thread. To assess the performance improvement introduced by DSPN, the total computation time T_t, which reflects the total computational amount, provides a more accurate measure. By comparing T_t, the proposed method can save between 8.68% (in Hamm) and 84.10% (in Arenella) of computation costs, as shown in Table 6.7. This demonstrates a strong correlation between the percentage of non-DSC and the speed-up ratio R_{sp} as defined in Eq. 6.9. Furthermore, this indicates that if only a single thread is used for preprocessing, the efficiency gain will closely align with the percentage of non-DSC.

$$R_{sp} = \frac{T_t(without\ DSPN) - T_t\ (DSPN)}{T_t(without\ DSPN)} \tag{6.9}$$

Table 6.7 The comparison between the percentage of Non-DSC and the speed up rate $\boldsymbol{R_{sp}}$ in test regions

Region	$\boldsymbol{Non-DSC}(\%)$	$\boldsymbol{R_{sp}}(\%)$
Wickede (Ruhr)	42.93	46.58
Münster	38.68	34.37
Hamm	9.12	11.74
Arenella	83.63	87.39
South side of Etna	11.28	13.3
Rocche d'Argimusco	23.86	25.79

6.4.2. Analysis of the False Negative samples

This study demonstrates that DSPN can generate a high-accuracy mask for predicting DSC after training. However, as the DS cannot be determined only by the threshold of γ_{PTA}, it is valuable to investigate what those False Negatives are like, and the potential reason that caused the misclassification. From the visual inspection of two optical images taken near the high coherence and low coherence acquisitions provided by Google Earth, it can be seen that most of the misclassification happened in the areas covered by deciduous and herbal plants such as fields, forests, street trees, etc. as shown in Fig. 6.12 and Fig. 6.13. Since the inputs of DSPN are from only two acquisitions, these two acquisitions usually correspond to different seasons, spring to summer or autumn to winter. This can lead to a situation where one acquisition yields better coherence than the other. The variation of radar reflection patterns of deciduous and herbal plants in different seasons is one of the potential causes of misclassification. Different growth and development stages of plants will change the amplitude and coherence of the polarized radar signal, which makes it possible for DSPN to make confused judgments on the plant coverage area.

After DS pre-processing, the displacement analysis is performed with help of the Stanford Method for PS (StaMPS). Those pixels found in fields or forests usually are filtered out through the selection step by StaMPS. By using the modified version of StaMPS from (Even et al., 2020) with the predicted mask, the error caused by FN can be evaluated by the percentage of lost coverage LC, which is calculated in the following way: Gridding the scene into approximately 200 m by 200 m cells, and counting the number C_{DS} of cells that contain DS for the result that was obtained without considering a mask and the number C_m of these cells, where DS were missing after a mask was applied. Then the percentage of lost coverage is:

$$LC = \frac{C_m}{C_{DS}} \tag{6.10}$$

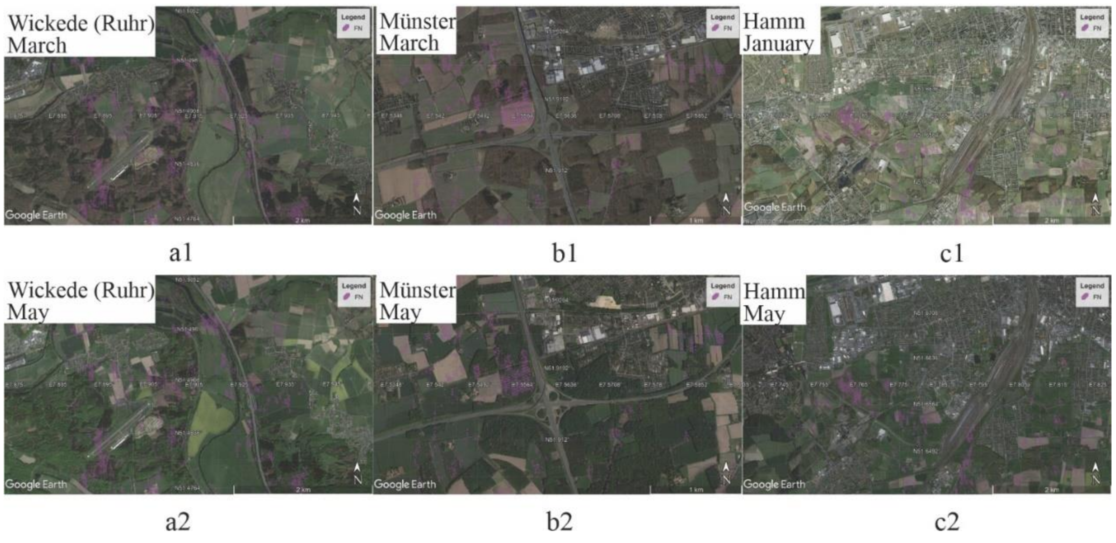


Fig. 6.12 False Negative points of test regions in North Rhine-Westphalia. (a1) Wickede (Ruhr) taken on March (a2) Wickede (Ruhr) taken on May (b1) Münster taken on March (b2) Münster taken on May. (c1) Hamm taken on January. (c2) Hamm taken on May.

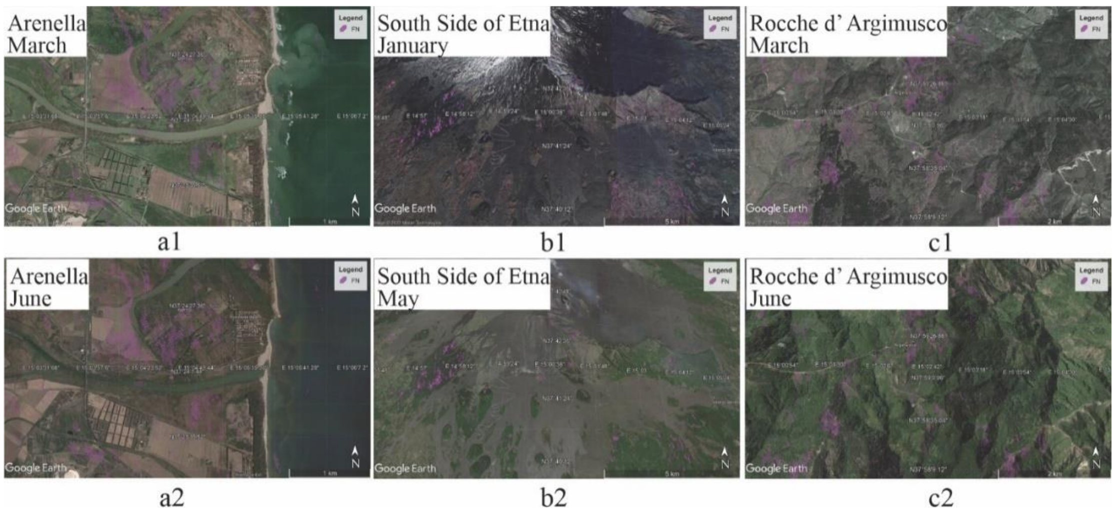


Fig. 6.13 False Negative points of test regions in Sicily. (a1) Arenella taken on March (a2) Arenella taken on June (b1) South side of Etna taken on January (b2) South side of Etna taken on May. (c1) Rocche d' Argimusco taken on March. (c2) Rocche d' Argimusco taken on June.

The lost coverage LC for the six test regions is shown in Table 6.8:

Table 6.8 Percentage of lost coverage caused by FN in test regions

	Wickede (Ruhr)	**Münster**	**Hamm**	**Arenella**	**South side of Etna**	**Rocche d'Argimusco**
LC(%)	3.0%	2.7%	0.6%	0.2%	0.5%	0.7%

6.4.3. Ablation study

Compared to the traditional U-Net, DSPN introduces improvements in both the network architecture and the loss function. To evaluate the performance enhancements resulting from these changes, this study compares the number of parameters, training time, GPU memory usage, and prediction time of DSPN and U-Net on the North Rhine-Westphalia dataset. The results of these comparisons are presented in Table 6.9. Additionally, using U-Net with H_{CE} as the baseline, the performance gains achieved by the DSPN network structure, and the H_{WCE} loss function are detailed in Table 6.10.

Table 6.9 Complexity comparison: DSPN VS. U-Net

	Parameters	**Training time**	**GPU memory needed**	**Prediction time**
DSPN	68003831	49h	6843.00MB/batch	56s
U-Net	51538466	27h	3382.60MB/batch	43s

Table 6.10 Ablation study results of North Rhine-Westphalia.

Region	**Network architecture**	**Loss function**	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**NPV (%)**	**FNR (%)**
Wickede (Ruhr)	DSPN	H_{WCE}	93.67	100	84.51	90.32	15.49
	DSPN	H_{CE}	87.55	99.98	69.58	82.61	30.42
	U-Net	H_{WCE}	75.44	72.13	67.85	77.61	32.14
	U-Net	H_{CE}	73.75	77.78	52.68	72.19	47.32
Münster	DSPN	H_{WCE}	94.69	100	87.22	91.68	12.78
	DSPN	H_{CE}	91.49	99.95	79.55	87.32	20.45
	U-Net	H_{WCE}	74.77	70.56	71.27	78.03	28.73
	U-Net	H_{CE}	74.85	76.57	60.11	73.97	39.89
Hamm	DSPN	H_{WCE}	98.41	100	97.91	93.82	2.09
	DSPN	H_{CE}	96.79	99.98	95.79	88.26	4.22
	U-Net	H_{WCE}	85.18	90.06	91.47	64.66	8.54
	U-Net	H_{CE}	83.83	92.64	86.55	58.34	13.45

From the ablation experiment results, DSPN's dense concatenation significantly enhances performance compared to U-Net. Due to DSPN's multi-path decoding structure, it exhibits a stronger capacity for detail perception, with much performance improvement attributed to a reduction in false positives, resulting from its more precise ability to capture details. In the case of traditional cross-entropy loss, each label is given equal weight, causing the network to treat the distinction between different DSC categories and between DSC and non-DSC categories with equal importance. However, with weighted loss, the network remains highly sensitive to misclassifications between DSC and non-DSC categories, while becoming less sensitive to misclassifications among different DSC labels. This focus on differentiating between DSC and non-DSC is likely enabled the network to better distinguish them than when using traditional loss functions. The primary benefit of the weighted coarse label cross-entropy comes from the reduction in false negatives, thereby decreasing the likelihood of type II errors.

Compared to U-Net, DSPN has 24.21% more parameters, which requires more time and memory for training. However, given the notable improvement in accuracy and the fact that prediction only takes a few minutes, the added complexity of the network is a justifiable cost. Furthermore, reducing false negatives will save time during the DS preprocessing phase. Additionally, since the weighted coarse label cross-entropy loss can be implemented using PyTorch functions, it benefits from CUDA-supported GPU parallel computing, ensuring that the use of this loss function does not significantly increase computation time.

6.4.4. Generalization ability

In general, DSPN training and predictions should be based on data from the same topographical region, such as different areas within the same image stack. However, as an extreme case, using predictions from a model trained on different topography can serve as a test of the network's generalization ability. This approach also introduces a potential application: the rapid generation of DSC previews from previously unseen image stacks. By employing models trained on regions with similar topography, users can obtain a preview of DSC distributions within just a few minutes, even when the topography and land cover types differ. Fig. 6.14 compares the predicted masks for Ibbenbüren, Germany, and Arenella, Italy, generated using the North Rhine-Westphalia model, alongside their corresponding ground truth. The results demonstrate that, despite the notable differences in land cover between Germany and Sicily, the overall distribution of DSC in both regions can still be identified. This property suggests that DSPN indeed learns the latent pattern of DS from the input features, rather than just "memorizing" the spatial distribution of DS (overfitting).

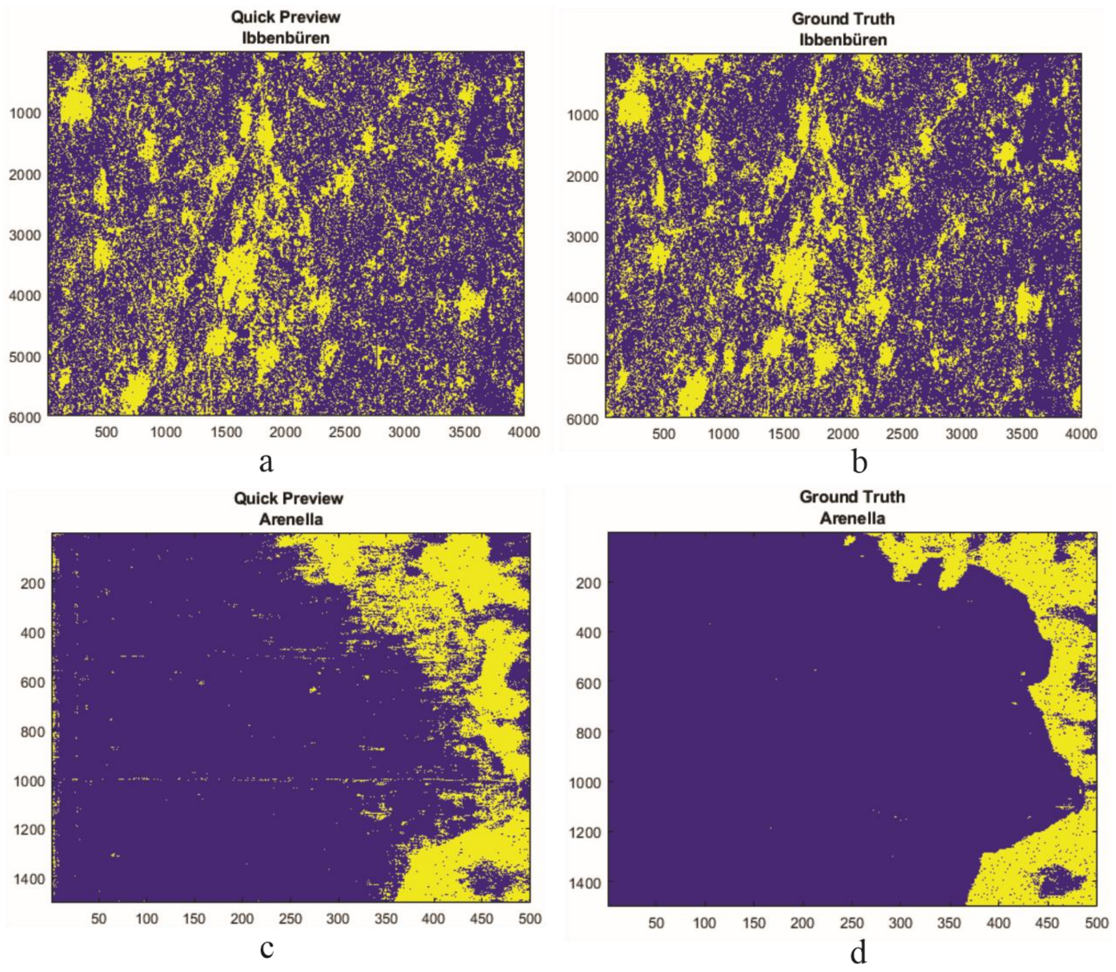


Fig. 6.14 Comparison between the quick previews and the ground truths. (a) Quick preview of Ibbenbüren by using North Rhine-Westphalia model. (b) Ground Truth of Ibbenbüren. (c) Quick preview of Arenella by using North Rhine-Westphalia model. (d) Ground Truth of Arenella.

6.5. Conclusion

This study proposed a DS preprocessing network called DSPN which utilizes deep learning technology, offering a novel method to accelerate DS preprocessing by predicting whether a pixel is classified as DSC prior to applying DS preprocessing. By only processing DSC predicted by DSPN, this method achieves a speed increase proportionate to the non-DSC ratio, without causing a significant loss in coverage. After testing across six different areas, this method demonstrated the ability to save between 11.74% and 87.39% of computation time, with a maximum coverage loss of only 3%. Integrating this approach into the InSAR deformation analysis processing chain can substantially enhance processing speed while maintaining nearly the same level of accuracy and coverage. Furthermore, this method enables the rapid preview of DS distribution in unknown image stacks, facilitating the selection of appropriate stacks and reducing unnecessary computation costs.

7. Conclusion and Outlook

The overall research theme of this thesis is to explore how deep learning can be used to solve modeling problems in geodetic applications. Specifically, the modeling methods of deep learning for temporal and spatial geodetic models are explored from the perspectives of tropospheric delay modeling problems and DS identification and prediction problems. This last chapter of the thesis provides a summary of the main conclusions and contributions of the study (Section 7.1), discusses general modeling methods and problem types where deep learning can be applied in geodesy (Section 7.2), and presents an outlook on the future of deep learning and artificial intelligence technologies within the field of geodesy (Section 7.3).

7.1. Summary and Contributions

In space geodetic applications, whether in GNSS-based positioning, water vapor retrieval, or InSAR-based ground motion monitoring, nonlinear terms such as atmospheric delay or coherence must be modeled and estimated. The performance of these models and estimations directly affects the accuracy and processing efficiency of geodetic applications. Therefore, determining effective modeling methods for these nonlinear terms has long been a focus of spaceborne geodetic research. This thesis proposes two deep learning-based solutions specifically for the challenges of tropospheric delay modeling, and DSC prediction in InSAR deformation monitoring. Compared to traditional approaches, these methods offer significant improvements in both accuracy and processing efficiency. The following is a summary of the key contributions and results of these approaches.

7.1.1. Deep learning model for tropospheric delay modeling

One of the main contributions of this study is the introduction of a new deep learning framework to address the tropospheric delay estimation problem that affects spaceborne geodesy (including both GNSS and InSAR) by combining deep learning with NWM ray tracing methods and surface empirical models. Traditionally, tropospheric delays are modeled using layered meteorological data, including temperature, air pressure, and water vapor pressure. When layered data is unavailable, surface meteorological data can be used for approximate modeling through empirical formulas. However, in-situ measurements of layered meteorological data are challenging to obtain, and even surface meteorological data still require specialized equipment to obtain, such as weather stations, which limits the access to accurate meteorological information. As an alternative, NWM provides coarse-resolution numerical simulations of meteorological data by assimilating and integrating information from multiple sensors with physical laws, resulting in estimates of surface and layered meteorological data in a grid format. By using interpolation methods, NWM grid data can be interpolated into specific locations and applied to estimate ZTD. However, due to uncertainties and accuracy loss during the assimilation and interpolation processes, ZTD estimates from NWM typically have an average error of nearly 10 mm when compared to direct GNSS measurements. Given that the accuracy of GNSS and InSAR deformation observations are at millimeter scale, the errors introduced by NWM-interpolated tropospheric delay have already had a non-negligible impact.

In this study, a deep learning framework named GM-LSTM is proposed to build a mapping from ZTD derived from NWM ray tracing to ZTD observed by GNSS to correct the errors introduced by assimilation and interpolation. It is important to clarify that the primary focus is not the inherent error of NWM itself—although NWM introduces uncertainty in the assimilation process, the ZTD error caused by imprecise meteorological data from NWM cannot be accurately measured due to the lack of in-situ measurement records and position for ZTD estimation is almost impossible to coincide with the grid position—but rather whether the total errors introduced by interpolation and NWM itself can be corrected. In this work, the integration of GNSS data is achieved not through traditional interpolation methods, but by using the ZWD time series observed from GNSS CORS network as supervisory data for the neural network. This supervision forces the neural network to adaptively learn the correction pattern of the ZWD time series estimated by NWM, thereby obtaining the ZTD correction. More importantly, this study is the first to treat the ZWD correction task as a probability density fitting task, rather than a numerical regression task. By applying GMM to describe ZWD corrected by GM-LSTM, this approach not only estimates the distribution of the corrected ZWD but also provides a measure of the correction uncertainty, which can be used to reflect the spatial heterogeneity caused by different weather activities at CORS stations and to evaluate the estimation quality. After testing in eight regions across Europe at different latitudes during both winter and summer, the proposed deep learning method, GM-LSTM, demonstrated superior performance. It reduced the RMSE by an average of 67.68%, 48.74%, and 49.63% compared to ZTD estimates from ERA5, VMF3, and GACOS, respectively. While traditional methods have an RMSE estimation error of approximately 10 mm, GM-LSTM achieved an RMSE of 4.52 mm in these eight test regions. This makes GM-LSTM a highly effective tool for estimating tropospheric delays at any location.

After obtaining ZTD from GNSS observations, it can be used to convert into IWV using meteorological data provided by NWM. With advances in GNSS receiver technology, some stations can generate real-time ZTD for real-time IWV retrieval. However, converting ZTD to IWV requires layered meteorological data, and in the absence of professional meteorological equipment, this conversion becomes a significant challenge. While NWM reanalysis products, such as ERA5, provide meteorological data accurate enough for real-time IWV retrieval, they cannot provide real-time data products that limit their application in real-time IWV retrieval scenarios. When reanalysis products are unavailable, traditional methods rely on empirical models like GPT3 to estimate meteorological parameters for real-time IWV retrieval, although their accuracy may not meet expectations.

To address this challenge and enable high-precision real-time IWV retrieval without relying on NWM reanalysis products, a deep learning method combining DNN and LSTM is proposed. This approach improves IWV retrieval accuracy by using real-time ZTD data and meteorological parameters estimated from the GPT3 empirical model. This deep learning method leverages a combination of DNN and LSTM to learn historical time series patterns during periods of high and low water vapor, thus enhancing real-time IWV retrieval results. In validation tests conducted at four GNSS stations located at different latitudes in Europe, the deep learning approach reduced the RMSE from nearly 4 $\mathrm{kg/m^2}$ to approximately 2-3 $\mathrm{kg/m^2}$, compared to real-time IWV retrieval using GPT3.

Although the task addressed by this method is different from that tackled by GM-LSTM, both models can be viewed as deep learning-based corrections to the tropospheric model derived from NWM meteorological data. In addition, given the similar temporal correlations between ZWD and IWV, both can be modeled as time series and features extracted using appropriate deep learning modules such as LSTM. Compared with traditional methods, deep learning-based tropospheric models not only utilize existing empirical models and NWM meteorological data, but also adopt data-driven techniques to capture the time series characteristics of the troposphere, thereby adaptively "enhancing" the model. From this perspective, deep learning methods are not opposed to traditional methods, but rather integrate and extend these methods through data-driven methods. By leveraging neural networks, deep learning provides an advanced fusion framework that improves the accuracy of the model. Deep learning models can extract temporal features from GNSS observation data (such as ZWD or IWV) to improve and correct the results of traditional methods, thereby improving overall accuracy. Compared to directly interpolating GNSS observations, this approach is adaptive and can more efficiently utilize existing observation archives without the need to manually adjust interpolation weight parameters. This adaptability and efficiency are one of the main advantages of deep learning in tropospheric delay modeling and is also the core contribution of this study.

7.1.2. Deep learning model for Distributed Scatterer Candidates prediction in InSAR deformation monitoring

In InSAR deformation analysis, deformation information is extracted by analyzing the phase difference between a master image and a slave image, referred to as the interferogram. However, the large time interval between these images can cause changes in the scattering properties of certain surface scatterers (e.g., variations in vegetation canopies), which can significantly affect the echo phase and lead to interferometric measurement failures. Consequently, identifying high-quality (high coherence) pixels from the interferogram stack becomes crucial for effective InSAR deformation analysis. In addition to data-driven modeling, using neural networks to extract features from a limited number of samples to estimate statistical metrics for the entire data stack is another application of deep learning in geodesy and remote sensing. Another important contribution of this study is the introduction of a deep learning method called DSPN that predicts distributed DSC in the entire image stack from statistical features derived from a pair of polarimetric SAR images.

Scatterers in InSAR deformation analysis are generally divided into two categories: PS and DS. PS exhibits natural phase stability (high coherence) and can be identified through filtering the interferogram and performing coherence screening, as in the StaMPS method. However, in rural areas, the number of scatterers that meet the PS criteria may be insufficient, resulting in phase unwrapping errors and reducing the reliability of deformation analysis. To address this limitation, statistically homogeneous DS are introduced alongside PS, providing more reliable deformation monitoring over a larger spatial extent. The inclusion of DS significantly increases the number of available pixels, enhancing the analysis range. However, this improvement comes at the cost of increased computational demands, as DS selecting requires estimating the spatial homogeneity of each pixel across the entire data stack. For large scenes, this process can be extremely time-consuming. To mitigate this challenge, this study proposes DSPN, which consists of a deep convolutional network to predict DSC using features extracted from two polarimetric SAR images. Since certain land cover types are highly unlikely to become DS, and the polarimetric SAR backscatter and coherence are complementary, it is reasonable to use polarimetric SAR images with the best and worst coherence to predict DS. However, the statistical characteristics of DS are typically described by the coherence of the whole data stack, making it challenging to capture the relationship between polarimetric SAR image features and DSC through traditional statistical or feature engineering methods. To address this, the proposed DSPN method transforms the DS prediction problem into a semantic segmentation task by utilizing coarse labels as supervisory information and employing a weighted cross-entropy loss function. The proposed DSPN convolutional neural network is designed to automatically extract relevant features from the input data, derived from polarimetric SAR images, to identify the pseudo-category of DSC. Following the correction of coarse labels and mask generation, the pseudo-category is then used to determine whether a pixel qualifies as a DSC. By focusing computational efforts only on predicted DSC, DSPN reduces the overall computational burden.

The significance of this work lies in its ability to filter out pixels that are unlikely to become DS before preprocessing, at minimal computational cost. Tests conducted in six different areas in North Rhine-Westphalia and Sicily showed that the proposed DSPN method has high accuracy in identifying and predicting DSC with almost no DS coverage loss (maximum coverage loss is about 3%). This allows DSPN to save computational time for DS preprocessing nearly to the DS ratio in the scene, with prediction times on the order of tens of seconds. These savings are particularly noticeable in rural or wilderness areas and water-covered regions. Across the six test areas, DSPN reduced computational time by 11.74% to 87.39%. In suburban areas such as Wickede (Ruhr), DSPN achieved nearly 47% computational time savings without sacrificing analysis accuracy. This deep learning approach is therefore highly significant for improving the efficiency of DS-InSAR processing.

Similar to GM-LSTM, DSPN is a complement to traditional methods rather than a replacement. DSC identified by DSPN still requires DS preprocessing methods to obtain optimal phase estimates. However, DSPN provides a low-cost solution to determine which pixels are worth further calculation. Neural networks are good at extracting high-level features from input data for classification (DSC identification), which is a very challenging task for manual feature engineering. Unlike GM-LSTM, which focuses on quantitative regression, DSPN applies deep learning to qualitative classification, representing another branch of deep learning methods in the field of geodesy.

7.2. What deep learning can do for geodesy

7.2.1. Where are we?

In recent years, the geodetic community has increasingly recognized the value of deep learning methods and has begun to explore their applications in various geodetic contexts. However, due to the domain knowledge gap between the traditional geodetic community and the artificial intelligence developer, current efforts from geodetic experts are mainly focused on adapting existing deep learning algorithms or toolkits to solve geodetic problems. In this context, it is often seen as intuitive to utilize MLP, ANN or DNN as a "generic regressor". The main advantage of this strategy is that under the premise of clear supervised data, it does not rely on model design for the problem, but expects the neural network to self-learn the pattern of the problem from the data. This allows the neural network to learn patterns inherent to the problem directly from the data. In addition, since there are no complex model architecture requirements, the number of hyperparameters that must be tuned is reduced and is usually limited to the network depth and number of neurons. This approach is often effective in scenarios with limited domain knowledge, as the best hyperparameter combination that achieves the best network performance can be quickly identified through grid search or similar techniques. Thus, trained neural networks can act as interpreters for geodetic models, mapping observed signals to variables of interest - such as converting ZWD data to IWV, or converting GNSS/SAR signals to environmental variables such as ionospheric total electron content, wind speed, and soil moisture, etc.

In addition, when the spatial and temporal characteristics of the data are well defined, combining domain-specific neural networks, such as RNN or LSTM for temporal data, and CNN for spatial data, can significantly improve performance and expand the scope of application. In temporal models, common use cases include exploiting the temporal correlation of geophysical variables to predict future data based on historical patterns. This can help GNSS or InSAR solutions by providing prior estimates, such as using past tropospheric delays or deformation time series to predict future states, thereby speeding up solutions and reducing errors in phase unwrapping. Besides, temporal networks can also facilitate outlier detection and data interpolation when clear time series patterns exist. For spatial models, typical applications include pixel-level feature detection—such as identifying PS or DS—and integrating and ensemble products of different resolutions. While there is an overlap with what spatial networks do in the remote sensing and computer vision field, spatial models in geodesy pose unique challenges due to the poorly defined problem specification. Therefore, choosing the right supervised data and learning methods is critical to the success of these models.

While these deep learning methods mentioned above that directly fit or regress geodetic problems may outperform traditional empirical models, they do not inherently incorporate geodetic or geophysical domain knowledge. As a result, neural networks act as "black boxes" and lack interpretability. This limitation effectively replaces traditional empirical models with opaque systems, hindering further performance improvements. A more important issue is the limited generalization capabilities of these black-box models—especially their inference capabilities. For example, models trained with regular data may not accurately respond to extreme meteorological events. Therefore, the need to integrate geodetic domain knowledge into deep learning frameworks has become imperative. Doing so aims to improve the generalization capabilities, interpretability, and overall accuracy of the models. Addressing this challenge is the core motivation and main contribution of this thesis.

7.2.2. What is a better approach for geodesy using deep learning?

As discussed in this thesis, deep learning, as an advanced data-driven approach, should not be viewed as a replacement for traditional geodetic methods but rather as an extension. The core step of this extension should be to ensure the perfect integration of geodetic domain knowledge with deep learning algorithms. It enables the use of data-driven techniques to develop models that perform better than manually designed models in geodetic applications. In terms of task categorization, the application of deep learning in geodesy can be broadly divided into two areas: qualitative tasks (classification, anomaly detection, recognition, segmentation, etc.) and quantitative tasks (regression, parameter fitting, etc.).

Qualitative tasks usually do not change the core geodetic processing workflow, but are introduced as supplementary pre-processing or post-processing steps. The purpose is to pre-screen the available data to improve processing efficiency or improve data quality. In some cases, qualitative tasks are also used to analyze processing results, such as identifying seasonal deformation trends or revealing the causes of deformation. When using deep learning for qualitative tasks, the challenge lies in defining suitable supervision methods. Unlike the fields of remote sensing and computer vision, qualitative tasks in geodesy are usually highly specialized and require domain knowledge to construct targeted training datasets. In some cases, the Ground truth used as supervisory information cannot be directly obtained through observation, so it is necessary to carefully select the model and design the loss function for the specific geodetic method. This thesis takes the DS identification and prediction problem as a representative of qualitative tasks and innovatively gives the pseudo-label generation method and the corresponding network structure and loss function improvement strategy. For this task, the performance of the model is determined by a whole set of solutions such as problem specification, dataset preparation, network structure design and training. The comprehensive selection of the most suitable overall solution constitutes the core view of this paper on deep learning to solve qualitative problems in geodetic measurement.

On the other hand, quantitative tasks involve integrating deep learning more directly into the geodetic processing chain, often used to estimate certain parameters that need to be modeled, such as ionospheric and tropospheric delays or multipath effects. In these cases, deep learning models aim to learn patterns from a limited set of direct observations that are constrained by temporal and spatial distribution. By leveraging historical observations or spatially adjacent data, these models can infer values of unknown times or locations, which can then be used to solve the observation equations. For such tasks, the challenge of modeling lies in choosing appropriate input features and neural network architectures to match the properties of the physical quantity being estimated. This paper takes the problem of tropospheric delay modeling as a representative of quantitative tasks, and innovatively presents methods for using neural networks to identify correction patterns from GNSS observation sequences and modeling ZWD using probability density. For such tasks, domain knowledge derived from geodesy can not only be introduced as input features of the network, but can also be used to guide the design of the network structure, which constitutes the core view of this paper on deep learning to solve geodetic quantitative problems.

7.3. Outlook and future work

In this thesis, deep learning-based methods are presented for tropospheric delay modeling (time-series modeling) and DSC prediction (spatial modeling). While these methods demonstrate significant potential, there remains room for further refinement and expansion. In the short term, the following explorations can be carried out in the future based on these two works:

1. Water Vapor Field Retrieval Based on GM-LSTM

Currently, GNSS-based water vapor retrieval provides only single-point measurements and is unable to generate large-scale water vapor fields. While multispectral remote sensing methods, such as MODIS, can offer large-scale water vapor field products, their accuracy is often influenced by factors such as cloud cover and day-night cycles. Additionally, the temporal resolution of these products is constrained by the satellite's revisit time. GM-LSTM has the capability to use GNSS observations to produce a ZWD model at any location within a region. This capability can be extended to generate a ZWD field at a specified spatial resolution, which can then be converted into IWV to refine the water vapor field obtained from NWM. Using this approach, it would be possible to produce regional water vapor field products with high temporal and spatial resolution, which would be valuable for applications requiring precise and timely earth observations, such as weather forecasting, disaster warnings, and precision agriculture.

2. InSAR Deformation Analysis Integrating GM-LSTM and DSPN

Since tropospheric delay is independent of microwave frequency, the ZTD field generated by GM-LSTM can also be applied for atmospheric correction in InSAR analysis. The ZTD field generated by GM-LSTM would serve as a more accurate ZTD product compared to traditional filtering methods (e.g., StaMPS step 8), the TRAIN toolbox (which uses ERA5 interpolation), or GACOS. By generating separate ZTD fields for the acquisition times of the master and slave images and then subtracting them, the phase contribution from atmospheric delay can be estimated and removed. Since the atmospheric delay phase from GM-LSTM is generated independently of the interferogram, this phase contribution can be subtracted during preprocessing, such as after DSPN predicts the available DSC. This approach has the advantage of separating spatially correlated deformation signals from atmospheric delay signals early in the process, thereby improving the accuracy of deformation analysis in subsequent filtering steps.

3. Globally applicable GM-LSTM tropospheric products

At present, GM-LSTM is still used as a regional model to correct the internal ZWD, but whether it is possible to use the global GNSS CORS station observation information to build a unified correction model needs further study. From the perspective of data feasibility, this approach is reasonable because the ZTD product obtained by NWM ray tracing still has global coverage, and there are currently more than 3,000 accessible CORS stations in the world. The challenge of this approach is how to introduce spatial information into the network to construct a unified network to correct the ZWD at any location in the world. In the future, the possibility of using longitude, latitude and altitude as geospatial information to expand GM-LSTM will be evaluated first, and global GNSS stations will be used for training. Subsequent models will be improved in a targeted manner based on the evaluation results.

In the medium to long term, dedicated deep learning algorithms for geospatial data are worth studying, for example:

1. Geospatial temporal data analysis methods based on graph neural networks

In geodetic applications, both InSAR and GNSS face the same problem in spatial distribution of data, that is, uneven geographical distribution. This makes it impossible for conventional CNN to directly process such spatial data. Graph neural networks (GNN), as a neural network with nodes and adjacency matrices as input, are expected to become a powerful analysis tool for unevenly distributed geospatial temporal data. This work will focus on the spatiotemporal unification of sites, which is not only beneficial for AI for Geodesy, but also can be used in related geospatial research, such as environmental research or meteorological research.

2. Using artificial intelligence algorithms to detect environmental variables from GNSS/InSAR observations

Since both GNSS and InSAR use microwaves as sensors, they have sensing capabilities for water molecules that other remote sensing methods do not have. Based on this physical basis, not only can atmospheric water be retrieved, but also the water content in vegetation and soil can be retrieved. The powerful nonlinear analysis capabilities of artificial intelligence combined with the high temporal resolution of SAR and the high spatial resolution of GNSS make it possible to detect environmental variables such as field-scale evapotranspiration, high-resolution soil moisture detection, and more accurate estimation of the urban heat island effect index.

Bibliography

Alduchov, O. A., & Eskridge, R. E. (1996). Improved Magnus form approximation of saturation vapor pressure. *Journal of Applied Meteorology*, *35*(4). https://doi.org/10.1175/1520-0450(1996)035<0601:IMFAOS>2.0.CO;2

Ansari, H., De Zan, F., & Bamler, R. (2017). Sequential Estimator: Toward Efficient InSAR Time Series Analysis. *IEEE Transactions on Geoscience and Remote Sensing*, *55*(10). https://doi.org/10.1109/TGRS.2017.2711037

Ansari, H., De Zan, F., & Bamler, R. (2018). Efficient Phase Estimation for Interferogram Stacks. *IEEE Transactions on Geoscience and Remote Sensing*, *56*(7). https://doi.org/10.1109/TGRS.2018.2826045

Arigony-Neto, J., Rau, F., Saurer, H., Jaña, R., Simões, J. C., & Vogt, S. (2007). A time series of SAR data for monitoring changes in boundaries of glacier zones on the Antarctic Peninsula. *Annals of Glaciology*, *46*. https://doi.org/10.3189/172756407782871387

Bandhauer, M., Isotta, F., Lakatos, M., Lussana, C., Båserud, L., Izsák, B., Szentes, O., Tveito, O. E., & Frei, C. (2022). Evaluation of daily precipitation analyses in E-OBS (v19.0e) and ERA5 by comparison to regional high-resolution datasets in European regions. *International Journal of Climatology*, *42*(2). https://doi.org/10.1002/joc.7269

Berardino, P., Fornaro, G., Lanari, R., & Sansosti, E. (2002). A new algorithm for surface deformation monitoring based on small baseline differential SAR interferograms. *IEEE Transactions on Geoscience and Remote Sensing*, *40*(11). https://doi.org/10.1109/TGRS.2002.803792

Bevis, M. (1994). GPS meteorology: mapping zenith wet delays onto precipitable water. *Journal of Applied Meteorology*, *33*(3). https://doi.org/10.1175/1520-0450(1994)033<0379:GMMZWD>2.0.CO;2

Bilitza, D., McKinnell, L. A., Reinisch, B., & Fuller-Rowell, T. (2011). The international reference ionosphere today and in the future. In *Journal of Geodesy* (Vol. 85, Issue 12). https://doi.org/10.1007/s00190-010-0427-x

Blewitt, G., Hammond, W., & Kreemer, C. (2018). Harnessing the GPS Data Explosion for Interdisciplinary Science. *Eos*, *99*. https://doi.org/10.1029/2018eo104623

Boehm, J., Werl, B., & Schuh, H. (2006). Troposphere mapping functions for GPS and very long baseline interferometry from European Centre for Medium-Range Weather Forecasts operational analysis data. *Journal of Geophysical Research: Solid Earth*, *111*(2). https://doi.org/10.1029/2005JB003629

Böhm, J., Salstein, D., Alizadeh, M. M., & Wijaya, D. D. (2013). Geodetic and Atmospheric Background. In J. Böhm & H. Schuh (Eds.), *Atmospheric Effects in Space Geodesy* (pp. 1–33). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-36932-2_1

Born, M., Wolf, E., & Hecht, E. (2000). Principles of Optics: Electromagnetic Theory of Propagation, Interference and Diffraction of Light . *Physics Today*, *53*(10). https://doi.org/10.1063/1.1325200

Boudouris, G. (1963). On the index of refraction of air, the absorption and dispersion of centimeterwaves by gases. *Journal of Research of the National Bureau of Standards, Section D: Radio Propagation*, *67D*(6). https://doi.org/10.6028/jres.067d.069

Bowhill, S. A. (1971). Introduction to ionospheric physics. *Journal of Atmospheric and Terrestrial Physics*, *33*(2). https://doi.org/10.1016/0021-9169(71)90209-1

Brunner, F. K., & Gu, M. (1991). An improved model for the dual frequency ionospheric correction of GPS observations. *Manuscripta Geodaetica*, *16*.

Brush, S. G. (1967). History of the Lenz-Ising model. *Reviews of Modern Physics*, *39*(4). https://doi.org/10.1103/RevModPhys.39.883

Caliñski, T., & Harabasz, J. (1974). A Dendrite Method Foe Cluster Analysis. *Communications in Statistics*, *3*(1). https://doi.org/10.1080/03610927408827101

Chen, C. W., & Zebker, H. A. (2002). Phase unwrapping for large SAR interferograms: Statistical segmentation and generalized network models. *IEEE Transactions on Geoscience and Remote Sensing*, *40*(8). https://doi.org/10.1109/TGRS.2002.802453

Choromanska, A., Henaff, M., Mathieu, M., Arous, G. Ben, & LeCun, Y. (2015). The loss surfaces of multilayer networks. *Journal of Machine Learning Research*, *38*.

Dach, R., Bockmann, E. (eds. . (2024). *International GNSS Service Technical Report 2023 (IGS Annual Report)*. https://doi.org/10.48350/179297

Dach, R., Lutz, S., Walser, P., & Fridez, P. (2015). User manual of the Bernese GNSS Software, Version 5.2. In *Astronomical Institute, University of Bern* (Vol. 3, Issue December 2015).

Dauphin, Y. N., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S., & Bengio, Y. (2014). Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *Advances in Neural Information Processing Systems*, *4*(January).

Di Giovanni, G., & Radicella, S. M. (1990). An analytical model of the electron density profile in the ionosphere. *Advances in Space Research*, *10*(11). https://doi.org/10.1016/0273-1177(90)90301-F

Ding, J., & Chen, J. (2020). Assessment of empirical troposphere model GPT3 based on NGL's global troposphere products. *Sensors (Switzerland)*, *20*(13). https://doi.org/10.3390/s20133631

Ding, J., Chen, J., Wang, J., & Zhang, Y. (2023). Characteristic differences in tropospheric delay between Nevada Geodetic Laboratory products and NWM ray-tracing. *GPS Solutions*, *27*(1). https://doi.org/10.1007/s10291-022-01385-2

Duchi, J., Hazan, E., & Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, *12*.

Durre, I., Williams, C. N., Yin, X., & Vose, R. S. (2009). Radiosonde-based trends in precipitable water over the Northern Hemisphere: An update. *Journal of Geophysical Research Atmospheres*, *114*(5). https://doi.org/10.1029/2008JD010989

Elliott, J. R., Walters, R. J., & Wright, T. J. (2016). The role of space-based observation in understanding and responding to active tectonics and earthquakes. In *Nature Communications* (Vol. 7). https://doi.org/10.1038/ncomms13844

Essen, L., & Froome, K. D. (1951). Dielectric constant and refractive index of air and its principal constituents at 24,000 Mc./s. *Nature*, *167*(4248). https://doi.org/10.1038/167512a0

Even, M. (2021). A study on algorithms and parameter settings for ds preprocessing. *International Geoscience and Remote Sensing Symposium (IGARSS)*, *2021-July*. https://doi.org/10.1109/IGARSS47720.2021.9553662

Even, M. (2022). A study on algorithms and parameter settings for ds preprocessing part 2. *IGARSS 2022 - 2022 IEEE International Geoscience and Remote Sensing Symposium*, 8–11. https://doi.org/10.1109/IGARSS46834.2022.9883345

Even, M., & Schulz, K. (2018). InSAR deformation analysis with distributed scatterers: A review complemented by new advances. *Remote Sensing*, *10*(5). https://doi.org/10.3390/rs10050744

Even, M., Westerhaus, M., & Simon, V. (2020). Complex Surface Displacements above the Storage Cavern Field at Epe, NW-Germany, Observed by Multi-Temporal SAR-Interferometry. *Remote Sensing*, *12*(20). https://doi.org/10.3390/rs12203348

Ferretti, A., Fumagalli, A., Novali, F., De Zan, F., Rucci, A., & Tebaldini, S. (2014). *Process for filtering interferograms obtained from SAR images acquired on the same area*. Google Patents.

Ferretti, A., Fumagalli, A., Novali, F., Prati, C., Rocca, F., & Rucci, A. (2011). A new algorithm for processing interferometric data-stacks: SqueeSAR. *IEEE Transactions on Geoscience and Remote Sensing*, *49*(9), 3460–3470. https://doi.org/10.1109/TGRS.2011.2124465

Ferretti, A., Monti-guarnieri, A., Prati, C., Rocca, F., & Massonnet, D. (2007). InSAR Principles: Guidelines for SAR Interferometry Processing and Interpretation. In *ESA Publications*.

Ferretti, A., Prati, C., & Rocca, F. (2001). Permanent scatterers in SAR interferometry. *IEEE Transactions on Geoscience and Remote Sensing*, *39*(1). https://doi.org/10.1109/36.898661

Franklin, J. (2005). The elements of statistical learning: data mining, inference and prediction. In *Mathematical Intelligencer* (Vol. 27, Issue 2). https://doi.org/10.1007/BF02985802

Gaddes, M. E., Hooper, A., & Bagnardi, M. (2019). Using Machine Learning to Automatically Detect Volcanic Unrest in a Time Series of Interferograms. *Journal of Geophysical Research: Solid Earth*, *124*(11). https://doi.org/10.1029/2019JB017519

Goldstein, R. M., & Werner, C. L. (1998). Radar interferogram filtering for geophysical applications. *Geophysical Research Letters*, *25*(21). https://doi.org/10.1029/1998GL900033

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning An MIT Press Book. In *Nature* (Vol. 29, Issue 7553).

Gou, J., & Soja, B. (2024). Global high-resolution total water storage anomalies from self-supervised data assimilation using deep learning algorithms. *Nature Water*, *2*(2). https://doi.org/10.1038/s44221-024-00194-w

Graves, A., & Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural Networks, 18(5–6). https://doi.org/10.1016/j.neunet.2005.06.042

Guarnieri, A. M., & Tebaldini, S. (2008). On the exploitation of target statistics for SAR interferometry applications. IEEE Transactions on Geoscience and Remote Sensing, 46(11). https://doi.org/10.1109/TGRS.2008.2001756

Gupta, M. R., Bengio, S., & Weston, J. (2014). Training highly multiclass classifiers. *Journal of Machine Learning Research*, *15*.

Hanssen, R. F. (2001). Radar Interferometry: Data Interpretation and Error Analysis - Ramon F. Hanssen - Google Books. In *Chemistry & …*.

Hargreaves, J. K. (2003). The solar-terrestrial environment - An introduction to geospace - the science of the terrestrial upper atmosphere, ionosphere and magnetosphere. In *Biochemical and Biophysical Research Communications* (Vol. 91, Issue 2).

Hartmann, G. K., & Leitinger, R. (1984). Range errors due to ionospheric and tropospheric effects for signal frequencies above 100 MHz. *Bulletin Géodésique*, *58*(2). https://doi.org/10.1007/BF02520897

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2016-Decem*, 770–778. https://doi.org/10.1109/CVPR.2016.90

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. *2015 IEEE International Conference on Computer Vision (ICCV)*, 1026–1034. https://doi.org/10.1109/ICCV.2015.123

Helliwell, R. A. (1966). Whistlers and Related Ionospheric Phenomena. *Geophysical Journal of the Royal Astronomical Society*, *11*(5). https://doi.org/10.1111/j.1365-246X.1966.tb03172.x

Herring, T. A. (1992). Modeling of atmospheric delay in the analysis space geodetic data. *Symposium on Refraction of Transatmospheric Signals in Geodesy*, *36*.

Hersbach, H., Bell, B., Berrisford, P., Biavati, G., Horányi, A., Sabater, J. M., Nicolas, J., Peubey, C., Radu, R., Rozum, I., & others. (2023). *ERA5 hourly data on pressure levels from 1940 to present, Tech. Rep.* Copernicus Climate Change Service (C3S). https://doi.org/https://doi.org/10.24381/cds.bd0915c6

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8). https://doi.org/10.1162/neco.1997.9.8.1735

Hooper, A. J. (2008). A multi-temporal InSAR method incorporating both persistent scatterer and small baseline approaches. *Geophysical Research Letters, 35*(16). https://doi.org/10.1029/2008GL034654

Hooper, A., Segall, P., & Zebker, H. (2007). Persistent scatterer interferometric synthetic aperture radar for crustal deformation analysis, with application to Volcán Alcedo, Galápagos. *Journal of Geophysical Research: Solid Earth*, *112*(7). https://doi.org/10.1029/2006JB004763

Hoque, M. M., & Jakowski, N. (2008). Estimate of higher order ionospheric errors in GNSS positioning. *Radio Science*, *43*(5). https://doi.org/10.1029/2007rs003817

Hoque, M. M., & Jakowski, N. (2010). Higher order ionospheric propagation effects on GPS radio occultation signals. *Advances in Space Research*, *46*(2). https://doi.org/10.1016/j.asr.2010.02.013

Hubel, D. H., & Wiesel, T. N. (1968). Receptive fields and functional architecture of monkey striate cortex. *The Journal of Physiology*, *195*(1). https://doi.org/10.1113/jphysiol.1968.sp008455

Jade, S., & Vijayan, M. S. M. (2008). GPS-based atmospheric precipitable water vapor estimation using meteorological parameters interpolated from NCEP global reanalysis data. *Journal of Geophysical Research Atmospheres*, *113*(3). https://doi.org/10.1029/2007JD008758

Janssen, V., Ge, L., & Rizos, C. (2004). Tropospheric corrections to SAR interferometry from GPS observations. *GPS Solutions*, *8*(3). https://doi.org/10.1007/s10291-004-0099-1

Jarlemark, P. O. J., & Emardson, T. R. (1998). Strategies for spatial and temporal extrapolation and interpolation of wet delay. *Journal of Geodesy*, *72*(6). https://doi.org/10.1007/s001900050174

Jiang, M., Ding, X., Hanssen, R. F., Malhotra, R., & Chang, L. (2015). Fast statistically homogeneous pixel selection for covariance matrix estimation for multitemporal InSAR. *IEEE Transactions on Geoscience and Remote Sensing*, *53*(3). https://doi.org/10.1109/TGRS.2014.2336237

Jiang, M., & Guarnieri, A. M. (2020). Distributed Scatterer Interferometry with the Refinement of Spatiotemporal Coherence. *IEEE Transactions on Geoscience and Remote Sensing*, *58*(6), 3977–3987. https://doi.org/10.1109/TGRS.2019.2960007

Jordan, M. I. (1997). Chapter 25 Serial order: A parallel distributed processing approach. In *Advances in Psychology* (Vol. 121, Issue C). https://doi.org/10.1016/S0166-4115(97)80111-2

Karl, T. R., & Trenberth, K. E. (2003). Modern Global Climate Change. In *Science* (Vol. 302, Issue 5651). https://doi.org/10.1126/science.1090228

Keskar, N. S., & Socher, R. (2017). *Improving Generalization Performance by Switching from Adam to SGD. 1*. http://arxiv.org/abs/1712.07628

Kestin, J., Sengers, J. V., Kamgar parsi, B., & Sengers, J. M. H. L. (1984). Thermophysical Properties of Fluid H2O. *Journal of Physical and Chemical Reference Data*, *13*(1). https://doi.org/10.1063/1.555707

Ketchen, D. J., & Shook, C. L. (1996). The application of cluster analysis in strategic management research: An analysis and critique. *Strategic Management Journal*, *17*(6). https://doi.org/10.1002/(sici)1097-0266(199606)17:6<441::aid-smj819>3.0.co;2-g

Kiani Shahvandi, M., Dill, R., Dobslaw, H., Kehm, A., Bloßfeld, M., Schartner, M., Mishra, S., & Soja, B. (2023). Geophysically Informed Machine Learning for Improving Rapid Estimation and Short-Term Prediction of Earth Orientation Parameters. *Journal of Geophysical Research: Solid Earth*, *128*(10). https://doi.org/10.1029/2023JB026720

King, M. D., Kaufman, Y. J., Menzel, W. P., & Tanré, D. (1992). Remote Sensing of Cloud, Aerosol, and Water Vapor Properties from the Moderate Resolution Imaging Spectrometer (MODIS). In *IEEE Transactions on Geoscience and Remote Sensing* (Vol. 30, Issue 1). https://doi.org/10.1109/36.124212

Klobuchar, J. A. (1986). DESIGN AND CHARACTERISTICS OF THE GPS IONOSPHERIC TIME DELAY ALGORITHM FOR SINGLE FREQUENCY USERS. *Record - IEEE PLANS, Position Location and Navigation Symposium*.

Klos, A., Bogusz, J., Pacione, R., Humphrey, V., & Dobslaw, H. (2023). Investigating temporal and spatial patterns in the stochastic component of ZTD time series over Europe. *GPS Solutions*, *27*(1). https://doi.org/10.1007/s10291-022-01351-y

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, *2*.

Landskron, D., & Böhm, J. (2018). VMF3/GPT3: refined discrete and empirical troposphere mapping functions. *Journal of Geodesy*, *92*(4). https://doi.org/10.1007/s00190-017-1066-2

Larry Gardner,Katherine Garcia-Sage, J. Y. (2024). *USU-GAIM*. https://ccmc.gsfc.nasa.gov/models/USU-GAIM~2.4.3/

LeCun, Y. (1988). A theoretical framework for Back-Propagation. In *Proceedings of the 1988 Connectionist Models Summer School*.

Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. https://doi.org/10.1038/nature14539

Liu, G., Huang, G., Xu, Y., Ta, L., Jing, C., Cao, Y., & Wang, Z. (2022). Accuracy Evaluation and Analysis of GNSS Tropospheric Delay Inversion from Meteorological Reanalysis Data. *Remote Sensing*, *14*(14). https://doi.org/10.3390/rs14143434

Löfgren, J. S., Björndahl, F., Moore, A. W., Webb, F. H., Fielding, E. J., & Fishbein, E. F. (2010). Tropospheric correction for InSAR using interpolated ECMWF data and GPS zenith total delay from the Southern California integrated GPS network. *International Geoscience and Remote Sensing Symposium (IGARSS)*. https://doi.org/10.1109/IGARSS.2010.5649888

Ma, L., Liu, Y., Zhang, X., Ye, Y., Yin, G., & Johnson, B. A. (2019). Deep learning in remote sensing applications: A meta-analysis and review. In *ISPRS Journal of Photogrammetry and Remote Sensing* (Vol. 152). https://doi.org/10.1016/j.isprsjprs.2019.04.015

Marini, J. W. (1972). Correction of Satellite Tracking Data for an Arbitrary Tropospheric Profile. *Radio Science*, *7*(2), 223–231. https://doi.org/10.1029/RS007I002P00223

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, *5*(4), 115–133. https://doi.org/10.1007/BF02478259/METRICS

Merchant, M. A., Obadia, M., Brisco, B., Devries, B., & Berg, A. (2022). Applying Machine Learning and Time-Series Analysis on Sentinel-1A SAR/InSAR for Characterizing Arctic Tundra Hydro-Ecological Conditions. *Remote Sensing*, *14*(5). https://doi.org/10.3390/rs14051123

Mestre-Quereda, A., Lopez-Sanchez, J. M., Vicente-Guijalba, F., Jacob, A. W., & Engdahl, M. E. (2020). Time-Series of Sentinel-1 Interferometric Coherence and Backscatter for Crop-Type Mapping. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *13*, 4070–4084. https://doi.org/10.1109/JSTARS.2020.3008096

Michaelides, R. J., Zebker, H. A., & Zheng, Y. (2019). An Algorithm for Estimating and Correcting Decorrelation Phase from InSAR Data Using Closure Phase Triplets. *IEEE Transactions on Geoscience and Remote Sensing*, *57*(12). https://doi.org/10.1109/TGRS.2019.2934362

Milletari, F., Navab, N., & Ahmadi, S. A. (2016). V-Net: Fully convolutional neural networks for volumetric medical image segmentation. *Proceedings - 2016 4th International Conference on 3D Vision, 3DV 2016*, 565–571. https://doi.org/10.1109/3DV.2016.79

Moon, W. M., Ristau, J., & Vachon, P. (1998). Feasibility of applying space-borne SAR interferometry for earthquake tectonic investigation. *Geosciences Journal*, *2*(2). https://doi.org/10.1007/BF02910486

Moreira, A., Prats-Iraola, P., Younis, M., Krieger, G., Hajnsek, I., & Papathanassiou, K. P. (2013). A tutorial on synthetic aperture radar. *IEEE Geoscience and Remote Sensing Magazine*, *1*(1). https://doi.org/10.1109/MGRS.2013.2248301

Moses, R. W. (2004). The High-Latitude Ionosphere and Its Effects on Radio Propagation. *Eos, Transactions American Geophysical Union*, *85*(19). https://doi.org/10.1029/2004eo190012

Nemirovski, A., Juditsky, A., Lan, G., & Shapiro, A. (2009). Robust Stochastic Approximation Approach to Stochastic Programming. *SIAM Journal on Optimization*, *19*(4), 1574–1609. https://doi.org/10.1137/070704277

Nikaein, T., Iannini, L., Molijn, R. A., & Lopez-Dekker, P. (2021). On the value of sentinel-1 insar coherence time-series for vegetation classification. *Remote Sensing*, *13*(16). https://doi.org/10.3390/rs13163300

Nilsson, T., Böhm, J., Wijaya, D. D., Tresch, A., Nafisi, V., & Schuh, H. (2013). *Path Delays in the Neutral Atmosphere*. https://doi.org/10.1007/978-3-642-36932-2_3

Odena, A., Dumoulin, V., & Olah, C. (2017). Deconvolution and Checkerboard Artifacts. *Distill*, *1*(10). https://doi.org/10.23915/distill.00003

Onn, F., & Zebker, H. A. (2006). Correction for interferometric synthetic aperture radar atmospheric phase artifacts using time series of zenith wet delay observations from a GPS network. *Journal of Geophysical Research: Solid Earth*, *111*(9). https://doi.org/10.1029/2005JB004012

Osah, S., Acheampong, A. A., Fosu, C., & Dadzie, I. (2021). Deep learning model for predicting daily IGS zenith tropospheric delays in West Africa using TensorFlow and Keras. *Advances in Space Research*, *68*(3). https://doi.org/10.1016/j.asr.2021.04.039

Pascanu, R., Mikolov, T., & Bengio, Y. (2013). On the difficulty of training recurrent neural networks. *30th International Conference on Machine Learning, ICML 2013*, *PART 3*.

Robert, C. (2014). Machine Learning, a Probabilistic Perspective. *CHANCE*, *27*(2). https://doi.org/10.1080/09332480.2014.914768

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9351*. https://doi.org/10.1007/978-3-319-24574-4_28

Rosenblatt, F. (1960). Perceptron Simulation Experiments. *Proceedings of the IRE*, *48*(3). https://doi.org/10.1109/JRPROC.1960.287598

Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, *20*(C). https://doi.org/10.1016/0377-

0427(87)90125-7

Rüeger, J. M. (2002). Refractive Index Formulae for Radio Waves. *FIG Technical Program*.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature 1986 323:6088*, *323*(6088), 533–536. https://doi.org/10.1038/323533a0

Russell, S., & Norvig, P. (2021). Artificial Intelligence A Modern Approach (4th Edition). In *Pearson Series*.

Saastamoinen, J. (1972). Atmospheric Correction for the Troposphere and Stratosphere in Radio Ranging Satellites. In *Geophysical Monograph Series* (pp. 247–251). https://doi.org/10.1029/gm015p0247

Samiei Esfahany, S. (2017). Exploitation of Distributed Scatterers in Synthetic Aperture Radar Interferometry. In *TU Delft University*.

Schaer, S. (1999). Mapping and Predicting the Earth's Ionosphere Using the Global Positioning System. *Dissertation of Astronomical Institute*.

Schindelegger, M., Böhm, S., Böhm, J., & Schuh, H. (2013). *Atmospheric Effects on Earth Rotation*. https://doi.org/10.1007/978-3-642-36932-2_6

Schröder, M., Lockhoff, M., Fell, F., Forsythe, J., Trent, T., Bennartz, R., Borbas, E., Bosilovich, M. G., Castelli, E., Hersbach, H., Kachi, M., Kobayashi, S., Robert Kursinski, E., Loyola, Di., Mears, C., Preusker, R., Rossow, W. B., & Saha, S. (2018). The GEWEX Water Vapor Assessment archive of water vapour products from satellite observations and reanalyses. *Earth System Science Data*, *10*(2). https://doi.org/10.5194/essd-10-1093-2018

Schunk, R. W., Scherliess, L., Sojka, J. J., Thompson, D. C., Anderson, D. N., Codrescu, M., Minter, C., Fuller-Rowell, T. J., Heelis, R. A., Hairston, M., & Howe, B. M. (2004). Global Assimilation of Ionospheric Measurements (GAIM). *Radio Science*, *39*(1). https://doi.org/https://doi.org/10.1029/2002RS002794

Seo, K. W., Ryu, D., Eom, J., Jeon, T., Kim, J. S., Youm, K., Chen, J., & Wilson, C. R. (2023). Drift of Earth's Pole Confirms Groundwater Depletion as a Significant Contributor to Global Sea Level Rise 1993–2010. In *Geophysical Research Letters* (Vol. 50, Issue 12). https://doi.org/10.1029/2023GL103509

Seymour, M. S., & Cumming, I. G. (1994). Maximum likelihood estimation for SAR interferometry. *International Geoscience and Remote Sensing Symposium (IGARSS)*, *4*, 2272–2274. https://doi.org/10.1109/igarss.1994.399711

Shi, J., Li, X., Li, L., Ouyang, C., & Xu, C. (2023). An Efficient Deep Learning-Based Troposphere ZTD Dataset Generation Method for Massive GNSS CORS Stations. *IEEE Transactions on Geoscience and Remote Sensing*, *61*. https://doi.org/10.1109/TGRS.2023.3276874

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*.

Steigenberger, P., Boehm, J., & Tesmer, V. (2009). Comparison of GMF/GPT with VMF1/ECMWF and implications for atmospheric loading. *Journal of Geodesy*, *83*(10). https://doi.org/10.1007/s00190-009-

0311-8

Stephens, M. A. (1970). Use of the Kolmogorov–Smirnov, Cramér–Von Mises and Related Statistics Without Extensive Tables. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, *32*(1). https://doi.org/10.1111/j.2517-6161.1970.tb00821.x

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *07-12-June-2015*. https://doi.org/10.1109/CVPR.2015.7298594

Tang, J., Ding, M., Yang, D., Fan, C., Khonsari, N., & Mao, W. (2024). Different data-driven prediction of global ionospheric TEC using deep learning methods. *International Journal of Applied Earth Observation and Geoinformation*, *130*, 103889. https://doi.org/https://doi.org/10.1016/j.jag.2024.103889

Teunissen, P. J. G., & Montenbruck, O. (2017). *Springer handbook of global navigation satellite systems* (Vol. 10). Springer.

Thayer, G. D. (1974). An improved equation for the radio refractive index of air. *Radio Science*, *9*(10). https://doi.org/10.1029/RS009i010p00803

Tikhomirov, V. M. (1991). On the Representation of Continuous Functions of Several Variables as Superpositions of Continuous Functions of a Smaller Number of Variables. In *Selected Works of A. N. Kolmogorov*. https://doi.org/10.1007/978-94-011-3030-1_55

Todorova, S., Hobiger, T., & Schuh, H. (2008). Using the Global Navigation Satellite System and satellite altimetry for combined Global Ionosphere Maps. *Advances in Space Research*, *42*(4). https://doi.org/10.1016/j.asr.2007.08.024

Trenberth, K. E., Dai, A., Rasmussen, R. M., & Parsons, D. B. (2003). The changing character of precipitation. In *Bulletin of the American Meteorological Society* (Vol. 84, Issue 9). https://doi.org/10.1175/BAMS-84-9-1205

Tucker, A. J., & Fannin, B. M. (1968). Analysis of ionospheric contributions to the Doppler shift of CW signals from artificial Earth satellites. *Journal of Geophysical Research*, *73*(13). https://doi.org/10.1029/ja073i013p04325

Vaquero-Martínez, J., Antón, M., Ortiz de Galisteo, J. P., Cachorro, V. E., Costa, M. J., Román, R., & Bennouna, Y. S. (2017). Validation of MODIS integrated water vapor product against reference GPS data at the Iberian Peninsula. *International Journal of Applied Earth Observation and Geoinformation*, *63*. https://doi.org/10.1016/j.jag.2017.07.008

Wang, D., Even, M., & Kutterer, H. (2022). Deep learning based distributed scatterers acceleration approach: Distributed scatterers prediction Net. In *International Journal of Applied Earth Observation and Geoinformation* (Vol. 115). https://doi.org/10.1016/j.jag.2022.103112

Wang, D., Yuan, P., Kutterer, H. (2024). Real-Time GNSS Integrated Water Vapor Sensing Based on Time Series Correction Deep Learning Models. In: International Association of Geodesy Symposia. Springer, Berlin, Heidelberg.

Wang, D., Wang, L., & Kutterer, H. (2025). An advanced tropospheric delay model based on Gaussian Mixed Long Short-Term Memory Network. IEEE Transactions on Geoscience and Remote Sensing.

Wicaksana, A., & Rachman, T. (2018). Pattern recognition and machine learning christopher M . Bishop. In *Angewandte Chemie International Edition, 6(11), 951–952.* (Vol. 3, Issue 1). https://medium.com/@arifwicaksanaa/pengertian-use-case-a7e576e1b6bf

Worden, J., Noone, D., Bowman, K., Beer, R., Eldering, A., Fisher, B., Gunson, M., Goldman, A., Herman, R., Kulawik, S. S., Lampel, M., Osterman, G., Rinsland, C., Rodgers, C., Sander, S., Shephard, M., Webster, C. R., & Worden, H. (2007). Importance of rain evaporation and continental convection in the tropical water cycle. *Nature*, *445*(7127). https://doi.org/10.1038/nature05508

Xiao, F., Tong, L., & Luo, S. (2019). A method for road network extraction from high-resolution SAR imagery using direction grouping and curve fitting. *Remote Sensing*, *11*(23). https://doi.org/10.3390/rs11232733

Xu, W. B., Li, Z. W., Ding, X. L., & Zhu, J. J. (2011). Interpolating atmospheric water vapor delay by incorporating terrain elevation information. In *Journal of Geodesy* (Vol. 85, Issue 9). https://doi.org/10.1007/s00190-011-0456-0

Yang, D., & Fang, H. (2023). Forecasting of global ionospheric TEC using a deep learning approach. *GPS Solutions*, *27*(2). https://doi.org/10.1007/s10291-023-01413-9

Yang, F., Guo, J., Zhang, C., Li, Y., & Li, J. (2021). A regional zenith tropospheric delay (Ztd) model based on gpt3 and ann. *Remote Sensing*, *13*(5). https://doi.org/10.3390/rs13050838

Yang, F., Meng, X., Guo, J., Yuan, D., & Chen, M. (2021). Development and evaluation of the refined zenith tropospheric delay (ZTD) models. *Satellite Navigation*, *2*(1). https://doi.org/10.1186/s43020-021-00052-0

Yu, C., Li, Z., & Penna, N. T. (2018). Interferometric synthetic aperture radar atmospheric correction using a GPS-based iterative tropospheric decomposition model. *Remote Sensing of Environment*, *204*(March 2017), 109–121. https://doi.org/10.1016/j.rse.2017.10.038

Yu, C., Li, Z., Penna, N. T., & Crippa, P. (2018). Generic Atmospheric Correction Model for Interferometric Synthetic Aperture Radar Observations. *Journal of Geophysical Research: Solid Earth*, *123*(10), 9202–9222. https://doi.org/10.1029/2017JB015305

Yu, C., Penna, N. T., & Li, Z. (2017). Generation of real-time mode high-resolution water vapor fields from GPS observations. *Journal of Geophysical Research*, *122*(3), 2008–2025. https://doi.org/10.1002/2016JD025753

Yuan, P., Hunegnaw, A., Alshawaf, F., Awange, J., Klos, A., Teferle, F. N., & Kutterer, H. (2021). Feasibility of ERA5 integrated water vapor trends for climate change analysis in continental Europe: An evaluation with GPS (1994–2019) by considering statistical significance. Remote Sensing of Environment, 260, 112416.

Zebker, H. A., & Goldstein, R. M. (1985). TOPOGRAPHIC MAPPING FROM INTERFEROMETRIC SYNTHETIC APERTURE RADAR OBSERVATIONS. *Digest - International Geoscience and Remote Sensing Symposium (IGARSS)*. https://doi.org/10.1029/jb091ib05p04993

Zeiler, M. D., & Fergus, R. (2014). Visualizing and Understanding Convolutional Networks arXiv:1311.2901v3 [cs.CV] 28 Nov 2013. *Computer Vision–ECCV 2014*, *8689*(PART 1), 818–833. https://doi.org/10.1007/978-3-319-10590-1_53

Zhang, Q., Kong, Q., Zhang, C., You, S., Wei, H., Sun, R., & Li, L. (2019). A new road extraction method using Sentinel-1 SAR images based on the deep fully convolutional neural network. *European Journal of Remote Sensing*, *52*(1). https://doi.org/10.1080/22797254.2019.1694447

Zhang, W., Gou, J., Möller, G., Zhang, S., Gao, Y., Wang, N., & Soja, B. (2024). A New Deep-Learning-Assisted Global Water Vapor Stratification Model for GNSS Meteorology: Validations and Applications. *IEEE Transactions on Geoscience and Remote Sensing*, *62*, 1–14. https://doi.org/10.1109/TGRS.2024.3479778

Zhao, K., Wulder, M. A., Hu, T., Bright, R., Wu, Q., Qin, H., Li, Y., Toman, E., Mallick, B., Zhang, X., & Brown, M. (2019). Detecting change-point, trend, and seasonality in satellite time series data to track abrupt changes and nonlinear dynamics: A Bayesian ensemble algorithm. *Remote Sensing of Environment*, *232*. https://doi.org/10.1016/j.rse.2019.04.034

Zhou, Z., Rahman Siddiquee, M. M., Tajbakhsh, N., & Liang, J. (2018). Unet++: A nested u-net architecture for medical image segmentation. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *11045 LNCS*, 3–11. https://doi.org/10.1007/978-3-030-00889-5_1

Zhu, X. X., Montazeri, S., Ali, M., Hua, Y., Wang, Y., Mou, L., Shi, Y., Xu, F., & Bamler, R. (2021). Deep Learning Meets SAR: Concepts, models, pitfalls, and perspectives. *IEEE Geoscience and Remote Sensing Magazine*, *9*(4). https://doi.org/10.1109/MGRS.2020.3046356

Zhu, X. X., Tuia, D., Mou, L., Xia, G.-S., Zhang, L., Xu, F., & Fraundorfer, F. (2017). *Deep learning in remote sensing: a review*. *December*. https://doi.org/10.1109/MGRS.2017.2762307

Zhu, Y., & Newell, R. E. (1994). Atmospheric rivers and bombs. *Geophysical Research Letters*, *21*(18). https://doi.org/10.1029/94GL01710

List of Figures

List of Tables

Acknowledgment

First and foremost, I would like to express my sincere gratitude to my supervisor, Prof. Dr.-Ing. Hansjörg Kutterer, for offering me the invaluable opportunity to take my first steps into the field of artificial intelligence for geodesy. His insightful comments and constant encouragement have been a cornerstone of my doctoral journey. Over the past four years, he has cultivated my ability to conduct independent scientific research and inspired me to continuously delve into the essence of scientific problems. This pursuit of depth and clarity has become a permanent asset in my academic career. In addition to providing me with academic guidance, Prof. Kutterer also sincerely cared about my life and the life of my wife in Germany during my stay. Without his encouragement and support, I believe it would have been difficult to persist in completing this work.

Furthermore, I would like to express my sincere gratitude to Prof. Dr. Benedikt Soja for his valuable review and insightful feedback on this thesis. I first had the pleasure of meeting Prof. Soja at the 2024 GGOS meeting, and his research on the application of artificial intelligence in geodesy has been a great source of inspiration for me. I am also deeply thankful to the members of the doctoral committee for their time and thoughtful evaluation of this work: Prof. Dr.-Ing. Hansjörg Kutterer, Prof. Dr. Benedikt Soja, Prof. Dr. Jan Cermak, and Prof. Dr.-Ing. Stefan Hinz.

I am particularly grateful to my colleague Dr. Markus Even, an expert in the field of InSAR and a guide for my research in geodesy. His support helped me understand the complex principles of InSAR and integrate AI into InSAR processing in a short period of time. His detailed guidance on coding during my first year of research laid the foundation for my interdisciplinary work, connecting geodesy and computer science. In addition, I am very grateful for his advice and help in data processing and his meticulous input in the writing and revision of the published paper and this thesis.

I am also very grateful to my former colleague Dr. Peng Yuan, whose collaboration opened a new chapter for me to explore AI for GNSS applications. I wish him great success in his career at GFZ.

I sincerely thank my colleague and friend Lingke Wang for his support in GNSS data processing and valuable suggestions for paper revision. I am deeply grateful for his care and help during my life in Karlsruhe.

I am lucky to work with a group of wonderful colleagues and former colleagues at GIK: Dr. Malte Westerhaus, Dr. Kurt Seitz, Dr. Michael Mayer, Dr. Thomas Grombein, Alison Larissa Seidel (and her dog Foxi), Charlotte Gschwind, Martina Pfersching, Jakob Weisgerber, Bence Ambrus, Lisa Dalheimer and Alexandra Duckstein. It has been a truly enriching and happy time to work with all of you during these four years.

I would like to thank the reviewers for the time they took to evaluate this thesis.

I would like to thank KHYS for providing funding through the Network Grant Program to enable me to participate in the academic exchange in Hong Kong. I would also like to thank GIK and KIT Library for their financial support for my participation in academic conferences and publications. I am especially grateful to the China Scholarship Council for its financial support during my stay in Germany.

Finally, I would like to express my special thanks to my family, especially my wife, Bingxiao Wen. Her unwavering support and encouragement are the driving force behind my doctoral research. In addition to helping with daily life, she is also a valuable advisor and listener, providing deep insights into my work. Her understanding and trust in my career path have kept me passionate about scientific research. I look forward to continuing our journey together and creating a better future.