ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

A UNIFIED DEEP LEARNING FRAMEWORK FOR SCALABLE LEARNING AND UNCERTAINTY QUANTIFICATION IN INTERVAL TYPE-2 FUZZY LOGIC SYSTEMS

M.Sc. THESIS

Ata KÖKLÜ

Department of Control and Automation Engineering

Control and Automation Engineering Programme

FEBRUARY 2026

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

A UNIFIED DEEP LEARNING FRAMEWORK FOR SCALABLE LEARNING AND UNCERTAINTY QUANTIFICATION IN INTERVAL TYPE-2 FUZZY LOGIC SYSTEMS

M.Sc. THESIS

Ata KÖKLÜ
(504231105)

Department of Control and Automation Engineering

Control and Automation Engineering Programme

Thesis Advisor: Prof. Dr. Tufan KUMBASAR

FEBRUARY 2026

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

ARALIK TİP-2 BULANIK MANTIK SİSTEMLERİNDE ÖLÇEKLENEBİLİR ÖĞRENME VE BELİRSİZLİK NİCELENDİRMESİ İÇİN DERİN ÖĞRENME TABANLI BİRLEŞTİRİLMİŞ BİR ÇERÇEVE

YÜKSEK LİSANS TEZİ

Ata KÖKLÜ
(504231105)

Kontrol ve Otomasyon Mühendisliği Anabilim Dalı

Kontrol ve Otomasyon Mühendisliği Programı

Tez Danışmanı: Prof. Dr. Tufan KUMBASAR

ŞUBAT 2026

Ata KÖKLÜ, a M.Sc. student of ITU Graduate School student ID 504231105 successfully defended the thesis entitled "A UNIFIED DEEP LEARNING FRAMEWORK FOR SCALABLE LEARNING AND UNCERTAINTY QUANTIFICATION IN INTERVAL TYPE-2 FUZZY LOGIC SYSTEMS", which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Prof. Dr. Tufan KUMBASAR** Istanbul Technical University	....
Jury Members :	**Assoc. Prof. Dr. İlker ÜSTOĞLU** Istanbul Technical University	....
	Dr. Alec STOTHERT Mathworks Inc.	....

Date of Submission : **31 December 2025**
Date of Defense : **03 February 2026**

For those who died alone,

FOREWORD

This thesis was prepared during my graduate studies at the Artificial Intelligence and Intelligent Systems Laboratory (AI2S). The research conducted within the scope of this thesis focuses on the design and training of fuzzy logic systems within deep learning frameworks. During the thesis period, I worked on topics including deep learning-based modeling, fuzzy system architectures, uncertainty-aware learning, prediction intervals, time-series forecasting, conformal prediction, and GPUbased parallel implementations. These studies contributed directly to the methodological and computational foundations of the proposed framework and supported the development of scalable and efficient learning structures presented in this thesis.

I would like to express my sincere gratitude to my research colleagues at AI2S for their technical discussions and collaborative environment, and to the laboratory director, Prof. Tufan Kumbasar, for his guidance and academic support throughout my studies.

I would also like to thank MathWorks for the scholarship support provided during my master's studies. This support was associated with a research project funded through a MathWorks Research Grant. I would like to specifically thank Dr. Alec Stothert and Dr. Rajibul Huq for their involvement in this project and for the professional interaction and technical insight they provided during this process.

February 2026

Ata KÖKLÜ
(Control and Automation Engineer)

TABLE OF CONTENTS

ABBREVIATIONS

BMM	:	Begian–Melek–Mendel
CPU	:	Central Processing Unit
CSCM	:	Center of Set Calculation Method
DL	:	Deep Learning
eHTSK	:	Enhanced HTSK
eHTSK2	:	Enhanced HTSK for IT2-FLSs
F2T	:	Failed to Train Experiments
FLS	:	Fuzzy Logic System
FOU	:	Footprint of Uncertainty
FPI	:	Failed to Learn PI Models
FS	:	Fuzzy Sets
GPU	:	Graphics Processing Unit
HTSK	:	High-Dimensional TSK
HTSK2	:	HTSK for IT2-FLSs
IQR	:	Inter Quartile Range
IT2	:	Interval Type-2
KM	:	Karnik-Mendel
KMA	:	Karnik-Mendel Algorithm
LMF	:	Lower Membership Function
MBS	:	Mini Batch Size
MF	:	Membership Function
MU	:	Memory Usage
MWNT	:	Multi-Weighted Nie-Tan
NaN	:	Not a Number
NP	:	Not Provided
NT	:	Nie-Tan
PI	:	Prediction Interval
PICP	:	Prediction Interval Coverage Probability
PINAW	:	Prediction Interval Normalized Average Width
PKM	:	Parametric Karnik-Mendel
RMSE	:	Root Mean Square Error
T1	:	Type-1
TR	:	Type-Reduced
TRS	:	Type-Reduced Set
TSK	:	Takagi-Sugeno-Kang
TT	:	Training Time
UMF	:	Upper Membership Function
WKM	:	Weighted Karnik-Mendel
WNT	:	Weighted Nie-Tan

SYMBOLS

$\boldsymbol{p}$	: Rule index, $p = 1, 2, \ldots, P$
$\boldsymbol{m}$	: Input index, $m = 1, 2, \ldots, M$
$\boldsymbol{n}$	: Sample index
$\boldsymbol{P}$	: Number of fuzzy rules
$\boldsymbol{M}$	: Number of input variables
$\boldsymbol{N}$	: Number of training samples
$\boldsymbol{B}$	: Mini-batch size (batch length)
$\boldsymbol{T}$	: Number of epochs
$\boldsymbol{t}$	: Epoch index
$\boldsymbol{\theta}$	: Learnable parameter set
$\boldsymbol{\theta}_{\boldsymbol{A}}$	: Antecedent parameter subset
$\boldsymbol{\theta}_{\boldsymbol{C}}$	: Consequent parameter subset
$\boldsymbol{\theta}_{M}$	: CSCM parameter subset (scalar form)
$\boldsymbol{\theta}_{\boldsymbol{M}}$	: CSCM parameter subset (vector form)
$\boldsymbol{C}$	: Feasible set of constraints for parameters
$\boldsymbol{L}$	: Left switching point in KMA
$\boldsymbol{R}$	: Right switching point in KMA
$\boldsymbol{x}$	: Input vector, $\boldsymbol{x} = (x_1, x_2, \ldots, x_M)^T$
$\boldsymbol{x_n}$	: n-th input sample
$\boldsymbol{x_m}$	: m-th component of the input vector
$\boldsymbol{x_{n,m}}$	: m-th component of $\boldsymbol{x}_n$
$\boldsymbol{x^{(i)}_{1:B}}$	: Mini-batch input block (samples 1 to B of batch i)
$\boldsymbol{y}$	: System output
$\boldsymbol{y_p}$	: Output of the p-th fuzzy rule
$\boldsymbol{y_n}$	: Ground-truth output for sample n
$\boldsymbol{y(x_n)}$	: Model prediction for sample n
$\boldsymbol{y_{T1}(x)}$	: Output of the T1-FLS
$\boldsymbol{y_{IT2}(x)}$	: Output of the IT2-FLS
$\boldsymbol{\underline{y}_{IT2}}$	: Lower bound of the TR IT2 output
$\boldsymbol{\overline{y}_{IT2}}$	: Upper bound of the TR IT2 output
$\boldsymbol{A_{p,m}}$	: T1 FS of the m-th input in the p-th rule
$\boldsymbol{\tilde{A}_{p,m}}$	: IT2 FS of the m-th input in the p-th rule
$\boldsymbol{\mu_{A_{p,m}}}$	: Membership function of $A_{p,m}$
$\boldsymbol{\underline{\mu}_{\tilde{A}_{p,m}}}$	: LMF
$\boldsymbol{\overline{\mu}_{\tilde{A}_{p,m}}}$	: UMF

$\boldsymbol{c_{p,m}}$	: Center of the Gaussian MF
$\boldsymbol{\sigma_{p,m}}$	: Standard deviation of the T1 Gaussian MF
$\boldsymbol{\underline{\sigma}_{p,m}}$	: Lower standard deviation of the IT2 Gaussian MF
$\boldsymbol{\overline{\sigma}_{p,m}}$	: Upper standard deviation of the IT2 Gaussian MF
$\boldsymbol{\tilde{\sigma}_{p,m}}$	: Interval-valued standard deviation, $\tilde{\sigma}_{p,m} = [\underline{\sigma}_{p,m}, \overline{\sigma}_{p,m}]$
$\boldsymbol{h_{p,m}}$	: Height of the LMF
$\boldsymbol{f_p}$	: Firing strength of the p-th rule (Type-1 FLS)
$\boldsymbol{\underline{f}_p}$	: Lower firing strength of the p-th rule (IT2-FLS)
$\boldsymbol{\overline{f}_p}$	: Upper standard deviation of the IT2 Gaussian MF
$\boldsymbol{a_{p,m}}$	: Linear consequent coefficient for input x_m
$\boldsymbol{a_{p,0}}$	: Bias term of the p-th rule consequent
$\boldsymbol{a}$	: Stacked consequent coefficients
$\boldsymbol{a_0}$	: Stacked bias vector
$\boldsymbol{\beta}$	: CSCM scalar parameter (WKM/WNT/BMM)
$\boldsymbol{\gamma_p}$	: CSCM weight for rule p (MWNT)
$\boldsymbol{\gamma}$	: CSCM weight vector, $\boldsymbol{\gamma} = \gamma_1, \ldots, \gamma_P)^T$
$\boldsymbol{\underline{v}_p}$	: PKM relaxed binary variable (lower)
$\boldsymbol{\overline{v}_p}$	: PKM relaxed binary variable (upper)
$\boldsymbol{\varepsilon_n}$	: Prediction error, $\varepsilon_n = y_n - y(\boldsymbol{x}_n)$
$\boldsymbol{\underline{\tau}}$	: Lower quantile level
$\boldsymbol{\overline{\tau}}$	: Upper quantile level
$\boldsymbol{\varphi}$	: Quantile pair, $\varphi = [\underline{\tau}, \overline{\tau}]$
I^{φ}	: Target prediction interval coverage (e.g., 99%)
$\boldsymbol{\theta^*}$	: Optimal parameter set
$\otimes$	: Tensor (Kronecker) product
$\odot$	: Element-wise multiplication
$\oplus$	: Element-wise addition
$\ominus$	: Element-wise subtraction
$\oslash$	: Element-wise division
$\mathbf{min}(\cdot)$	: Minimum operator
$\mathbf{max}(\cdot)$	: Maximum operator

LIST OF TABLES

LIST OF FIGURES

Page

A UNIFIED DEEP LEARNING FRAMEWORK FOR SCALABLE LEARNING AND UNCERTAINTY QUANTIFICATION IN INTERVAL TYPE-2 FUZZY LOGIC SYSTEMS

SUMMARY

Fuzzy Logic Systems are widely used in control, prediction, and modeling problems where uncertainty, nonlinearity, and complex input–output relationships are present. Their rule-based structure provides high interpretability, which makes them attractive for engineering applications. Type-1 Fuzzy Logic Systems have been successfully applied to many practical problems due to their simple and fixed structure. However, real-world data often include measurement noise, model uncertainty, and environmental variability. Because Type-1 Fuzzy Logic Systems use fixed membership functions, they cannot represent such uncertainties adequately, which limits their performance and applicability.

To address these limitations, Interval Type-2 Fuzzy Logic Systems were introduced. In these systems, membership degrees are defined as intervals rather than single values through the use of the footprint of uncertainty. This structure allows uncertainty to be modeled in a more flexible and realistic manner. In regression problems, generating prediction intervals instead of only point predictions is especially important for safety-critical and high-risk applications. Interval Type-2 Fuzzy Logic Systems naturally support the generation of prediction intervals while preserving model interpretability.

Despite these advantages, the practical use of Interval Type-2 Fuzzy Logic Systems is limited by high computational cost and poor scalability. Uncertainty processing relies on center of sets calculation methods, which are often iterative and require sorting operations. Methods such as the Karnik–Mendel Algorithm are computationally expensive and not suitable for parallel computation. As a result, training and inference become slow, particularly for large data sets and high-dimensional input spaces.

Many existing learning methods for Interval Type-2 Fuzzy Logic Systems are direct extensions of methods developed for Type-1 Fuzzy Logic Systems. In these approaches, uncertainty-related parameters are treated as additional trainable variables, and learning mainly focuses on improving point prediction accuracy. Consequently, uncertainty quantification quality is not directly optimized, and prediction intervals may be unreliable. In addition, these methods do not effectively utilize modern machine learning tools such as mini-batch training, automatic differentiation, and graphics processing unit acceleration.

The main objective of this thesis is to develop a unified learning framework that embeds Type-1 and Interval Type-2 Fuzzy Logic Systems into modern deep learning environments. The proposed framework aims to reduce computational cost, address high-dimensional input spaces, and directly optimize uncertainty information during learning. To achieve this, fuzzy system structures are reformulated using appropriate parameterization strategies, allowing unconstrained optimization and efficient mini-batch training.

New inference formulations are proposed to remove the iterative structure of traditional center of sets calculation methods. All inference and learning operations are expressed in a fully vectorized and parallelizable form, enabling efficient execution on graphics processing units. Experimental results show that the proposed framework significantly reduces training and inference time while maintaining prediction accuracy and improving uncertainty estimation.

In conclusion, this thesis presents an efficient and scalable learning framework that integrates fuzzy logic systems with modern deep learning tools. The proposed methods preserve the interpretability of fuzzy rule-based models while enabling reliable uncertainty modeling and practical applicability in large-scale and high-dimensional problems.

ARALIK TİP-2 BULANIK MANTIK SİSTEMLERİNDE ÖLÇEKLENEBİLİR ÖĞRENME VE BELİRSİZLİK NİCELENDİRMESİ İÇİN DERİN ÖĞRENME TABANLI BİRLEŞTİRİLMİŞ BİR ÇERÇEVE

ÖZET

Bulanık Mantık Sistemleri, belirsizlik, doğrusal olmayan ve karmaşık giriş çıkış ilişkilerinin söz konusu olduğu problemlerde güçlü modelleme kabiliyetleri sunmaları nedeniyle kontrol, kestirim ve modelleme gibi alanlarda yaygın olarak kullanılmaktadır. Kural tabanlı yapıları sayesinde yüksek yorumlanabilirlik sağlamaları, bulanık mantık sistemlerini özellikle mühendislik uygulamaları açısından cazip kılmaktadır. Tip-1 Bulanık Mantık Sistemleri, sabit yapılarıyla birçok pratik problemde başarılı sonuçlar vermiştir. Buna rağmen gerçek dünya verilerinin doğasında bulunan ölçüm gürültüsü, model belirsizliği ve çevresel değişkenlik gibi unsurlar Tip-1 Bulanık Mantık Sistemlerinin belirsizliği yeterince temsil edememesi nedeniyle performanslarını ve kullanım alanlarını kısıtlamaktadır.

Bu sınırlamaların aşılması amacıyla geliştirilen Aralık Değerli Tip-2 Bulanık Mantık Sistemleri, üyelik derecelerinin tek bir değerden ziyade bir aralık içerisinde tanımlanmasına olanak tanıyan belirsizlik ayak izi (footprint of uncertainty) kavramını kullanmaktadır. Bu sayede Aralık Değerli Tip-2 Bulanık Mantık Sistemleri belirsizliğin daha gerçekçi ve esnek bir şekilde modellenmesini mümkün kılmaktadır. Özellikle regresyon problemlerinde, yalnızca noktasal tahminler yerine tahmin aralıklarının (prediction intervals) üretilmesi, güvenlik kritik ve yüksek risk içeren uygulamalarda büyük önem taşımaktadır. Aralık Değerli Tip-2 Bulanık Mantık Sistemleri, yapıları gereği tahmin aralıklarının üretimi için doğal ve yorumlanabilir bir çerçeve sunmaktadır.

Buna karşın, Aralık Değerli Tip-2 Bulanık Mantık Sistemlerinin pratik kullanımı, yüksek hesaplama maliyeti ve ölçeklenebilirlik sorunları nedeniyle ciddi biçimde kısıtlanmaktadır. Aralık Değerli Tip-2 Bulanık Mantık Sistemlerinde belirsizlik modelleme süreci, küme merkezi hesaplama yöntemleri (Center of Sets Calculation Methods) aracılığıyla gerçekleştirilmektedir. Karnik–Mendel Algoritması gibi yaygın küme merkezi hesaplama yöntemleri, iteratif ve sıralama algoritmaları içeren yapıları nedeniyle hesaplama açısından maliyetlidir ve paralel hesaplama mimarilerine uygun değildir. Bu durum, özellikle büyük veri kümeleri ve yüksek sayıda girişe sahip sistemlerde hem eğitim hem de çıkarım süreçlerini belirgin biçimde yavaşlatmaktadır.

Literatürde önerilen birçok Aralık Değerli Tip-2 Bulanık Mantık Sistemi öğrenme yöntemi, esasen Tip-1 Bulanık Mantık Sistemi öğrenme yaklaşımlarının doğrudan uzantıları niteliğindedir. Bu yöntemlerde belirsizlikle ilişkili parametreler çoğunlukla ek öğrenilebilir değişkenler olarak ele alınmakta ve öğrenme süreci büyük ölçüde noktasal tahmin doğruluğunu artırmaya odaklanmaktadır. Bu nedenle, belirsizlik modellemesinin kalitesi doğrudan hedeflenmemekte ve elde edilen tahmin aralıkları

çoğu zaman yetersiz ve/veya güvensiz kalmaktadır. Ek olarak bu yaklaşımlar modern makine öğrenmesi uygulamalarında yaygın olarak kullanılan mini-batch tabanlı optimizasyon, otomatik türev alma ve grafik işlemci hızlandırma gibi olanaklardan yeterince yararlanamamaktadır. Ayrıca, mevcut çalışmaların büyük bir kısmı sınırlı veri kümeleri üzerinde değerlendirilmiş olup, büyük ölçekli ve yüksek boyutlu problemlerde ortaya çıkan hesaplama ve belirsizlik modelleme sorunlarını yeterince ele almamaktadır. Literatürde önerilen yöntemlerin önemli bir bölümü, belirsizlik bilgisini model çıktılarına doğrudan yansıtmaktan ziyade, dolaylı biçimde ele almakta ve bu durum tahmin aralıklarının güvenilirliğini olumsuz etkilemektedir. Bu eksiklikler, Aralık Değerli Tip-2 Bulanık Mantık Sistemlerinin pratik ve endüstriyel uygulamalarda yaygın biçimde kullanılmasının önünde önemli bir engel oluşturmaktadır.

Bu tez çalışmasının temel amacı, Aralık Değerli Tip-2 Bulanık Mantık Sistemlerini uygun parametreleştirme yaklaşımları ile modern derin öğrenme yapıları içerisine gömerek, geleneksel öğrenme süreçlerinde karşılaşılan kısıtları ortadan kaldırmak ve mini-batch tabanlı derin öğrenme optimizasyon algoritmalarının etkin biçimde kullanılmasını sağlamaktır. Bu doğrultuda tez, küme merkezi hesaplama yöntemlerinin ve yüksek boyutlu Takagi–Sugeno–Kang yapılarının yeniden tasarlanması yoluyla çıkarım karmaşıklığını önemli ölçüde azaltmayı ve belirsizlik modelleme performansını iyileştirmeyi hedeflemektedir. Ayrıca, Aralık Değerli Tip-2 Bulanık Mantık Sistemlerine ait çıkarım süreçlerinin tamamen vektörleştirilmiş ve paralelleştirilebilir bir biçimde yeniden formüle edilmesiyle, geleneksel küme merkezi hesaplama yöntemlerinin iteratif yapısının ortadan kaldırılması ve grafik işlemci mimarileri üzerinde verimli biçimde çalıştırılması amaçlanmaktadır. Grafik işlemci tabanlı paralel hesaplamanın derin öğrenme çerçeveleri içerisinde etkin biçimde kullanılması sayesinde, tahmin doğruluğu veya belirsizlik modelleme kabiliyetinden ödün vermeden eğitim ve çıkarım sürelerinin önemli ölçüde hızlandırılması hedeflenmektedir. Son olarak, noktasal tahmin doğruluğu ile tahmin aralığı kalitesini eş zamanlı olarak optimize eden çift odaklı bir öğrenme stratejisinin, özellikle yüksek boyutlu ve büyük ölçekli problemler için geleneksel Aralık Değerli Tip-2 Bulanık Mantık Sistemi öğrenme yaklaşımlarına kıyasla üstün performans sunacağı tez kapsamında ortaya konulmaktadır.

Tez kapsamında öncelikle Tip-1 Bulanık Mantık Sistemlerinin ve Aralık Değerli Tip-2 Bulanık Mantık Sistemlerinin matematiksel yapıları, derin öğrenme ortamlarıyla uyumlu olacak şekilde yeniden formüle edilmiştir. Geleneksel bulanık öğrenme süreçlerinde karşılaşılan parametre kısıtları, uygun stratejileri kullanılarak kısıtsız optimizasyon problemlerine dönüştürülmüştür. Bu dönüşüm sayesinde standart derin öğrenme optimizasyon algoritmaları (örneğin Adam ve Stokastik Gradyan İnişi) ve otomatik türev alma mekanizmaları doğrudan kullanılabilir hale gelmiştir. Ayrıca, çıkarım ve öğrenme süreçleri mini-batch uyumlu olacak şekilde tanımlanmış ve tüm hesaplamalar matris tabanlı, vektörleştirilmiş ifadeler üzerinden gerçekleştirilmiştir.

Aralık Değerli Tip-2 Bulanık Mantık Sistemleri için, Karnik–Mendel Algoritmasının iteratif yapısını ortadan kaldıran ve tamamen paralelleştirilebilir yeni çıkarım yöntemleri geliştirilmiştir. Önerilen yaklaşım, tip indirgeme sürecinde oluşabilecek tüm olası kombinasyonları eş zamanlı olarak değerlendirebilmekte ve böylece grafik işlemci tabanlı paralel hesaplamadan etkin biçimde yararlanmaktadır. Deneysel sonuçlar, bu yöntemin Karnik–Mendel Algoritması tabanlı çıkarıma kıyasla eğitim süresini önemli ölçüde azalttığını ve tahmin doğruluğunu koruduğunu göstermektedir.

Tezin önemli katkılarından biri, Aralık Değerli Tip-2 Bulanık Mantık Sistemlerinin belirsizlik modelleme kabiliyetlerini artırmaya yönelik metodolojik geliştirmelerdir. Bu kapsamda literatürde yer alan küme merkezi hesaplama yöntemleri derin öğrenme ortamlarına uyarlanmış ve yeni ağırlıklı ve parametrik küme merkezi hesaplama yapıları önerilmiştir. Özellikle Parametrik Karnik–Mendel yöntemi, belirsizlik modelleme kabiliyeti, noktasal doğruluk ve hesaplama verimliliği arasında dengeli bir performans sunmuştur. Ayrıca, belirsizlik ve doğruluğun eş zamanlı olarak optimize edildiği çift odaklı öğrenme stratejisi geliştirilmiş ve bu yaklaşımın tahmin aralığı kalitesini anlamlı ölçüde artırdığı gösterilmiştir.

Boyutluluk laneti problemi (The Curse of Dimensionality), özellikle yüksek boyutlu giriş uzaylarında kural ateşleme güçlerinin hızla sıfıra yaklaşması şeklinde ortaya çıkmaktadır. Bu tezde, bu problemin üstesinden gelmek amacıyla yüksek boyutlu Takagi–Sugeno–Kang yapıları ve değiştirilmiş kural ateşleme mekanizmaları incelenmiştir. HTSK, HTSK2, eHTSK ve eHTSK2 yapıları, softmax tabanlı normalizasyon kullanarak kural aktivasyonlarını yeniden tanımlamakta ve yüksek boyutlu problemlerde daha kararlı çıkarım sağlamaktadır. Deneysel sonuçlar, geliştirilmiş eHTSK2 yapısının hem Tip-1 Bulanık Mantık Sistemlerinde hem de Aralık Değerli Tip-2 Bulanık Mantık Sistemlerinde en iyi genel performansı sağladığını göstermektedir.

Tez kapsamında gerçekleştirilen kapsamlı deneysel çalışmalar, önerilen methotların büyük ölçekli ve yüksek boyutlu veri kümeleri üzerinde geleneksel yöntemlere kıyasla belirgin üstünlükler sağladığını ortaya koymaktadır. Noktasal tahmin doğruluğunun yanında, tahmin aralığı kapsama oranı ve aralık genişliği gibi belirsizlik ölçütlerinde de anlamlı iyileşmeler elde edilmiştir. Bu sonuçlar, önerilen yaklaşımın belirsizlik farkındalığının kritik olduğu uygulamalar için uygun bir çözüm sunduğunu göstermektedir. Tezde önerilen öğrenme çerçevesi, yalnızca teorik bir yeniden formülasyon sunmakla kalmamakta, aynı zamanda pratik uygulamalara doğrudan aktarılabilecek hesaplamasal araçlar da sağlamaktadır. Geliştirilen derin öğrenme uyumlu yapı sayesinde, Bulanık Mantık Sistemleri modern makine öğrenmesi ekosistemlerinin sunduğu temel bileşenlerle bütünleşik biçimde eğitilebilir hale gelmiştir. Otomatik türev alma, grafik işlemci hızlandırma ve mini-batch tabanlı eğitim mekanizmalarının etkin biçimde kullanılması, Bulanık Mantık Sistemlerinin büyük ölçekli veri problemlerinde uygulanabilirliğini artırmıştır. Deneysel değerlendirmeler, farklı veri ölçekleri ve giriş boyutları dikkate alınarak sistematik biçimde gerçekleştirilmiştir. Bu süreçte, eğitim ve çıkarım süreleri, model ölçeklenebilirliği ve belirsizlik ölçütleri birlikte ele alınmış; önerilen yöntemlerin farklı problem senaryolarındaki davranışları ayrıntılı biçimde analiz edilmiştir. Elde edilen sonuçlar, yalnızca mutlak performans değerleri üzerinden değil, aynı zamanda yöntemlerin artan veri boyutu karşısındaki kararlılığı açısından da değerlendirilmiştir.

Sonuç olarak bu tez, Tip-1 Bulanık Mantık Sistemlerinin ve Aralık Değerli Tip-2 Bulanık Mantık Sistemlerinin modern derin öğrenme altyapılarıyla bütünleştirilmesine yönelik kapsamlı ve sistematik bir çerçeve sunmaktadır. Önerilen yöntemler, Bulanık Mantık Sistemlerinin yorumlanabilir kural tabanlı yapısını korurken, grafik işlemci destekli ölçeklenebilirlik, hesaplama verimliliği ve yüksek kaliteli belirsizlik modelleme yetenekleri kazandırmaktadır. Elde edilen bulgular, Aralık Değerli Tip-2 Bulanık Mantık Sistemlerinin büyük veri ve yüksek boyutlu regresyon problemlerinde hem doğruluk hem de güvenilir belirsizlik tahmini gerektiren uygulamalar için güçlü ve pratik modeller olduğunu ortaya koymaktadır.

Gelecek çalışmalar kapsamında, önerilen öğrenme çerçevesinin farklı derin öğrenme kütüphanelerine (örneğin PyTorch) uyarlanması, TinyML tabanlı gömülü sistemlerde kullanılabilir hale getirilmesi ve genel Tip-2 Bulanık Mantık Sistemlerini destekleyecek şekilde genişletilmesi planlanmaktadır.

1. INTRODUCTION

Type-1 (T1) and Interval Type-2 (IT2) Fuzzy Logic Systems (FLSs) are widely used in modeling, control, and learning tasks due to their ability to handle nonlinearity and uncertainty [1,2]. In particular, IT2-FLSs are powerful tools for representing uncertainty, as they employ IT2 Fuzzy Sets (FSs) as membership functions (MFs), enabling an additional degree of freedom through the footprint of uncertainty [3,4]. The effectiveness of IT2-FLSs in managing uncertainty strongly depends on their inference mechanisms and structural design, which determine how uncertainty is propagated and quantified [1,5]. In regression problems, uncertainty quantification is often expressed through Prediction Intervals (PIs), which provide interval-valued predictions instead of single point estimates and are particularly important in high-risk and safety-critical applications [6]–[8]. Due to their inherent ability to model uncertainty via the footprint of uncertainty, IT2-FLSs offer a natural and interpretable framework for PI generation.

Uncertainty processing in IT2-FLSs primarily relies on Center of Sets Calculation Methods (CSCMs) [5]. Among these, the Karnik-Mendel Algorithm (KMA) is one of the most commonly used approaches; however, its iterative and sorting based nature results in high computational complexity and long training and inference times [9]. As a result, conventional CSCMs limit parallel execution and reduce scalability in large-scale learning settings. This limitation becomes more pronounced when learning from large-scale datasets and contributes to curse of dimensionality issues [10]–[13].

Most learning methods for IT2-FLSs remain direct extensions of T1-FLS learning approaches [3,14], which often treat uncertainty related parameters merely as additional learnable parameters and focus primarily on point wise accuracy. Consequently, the quality of the uncertainty estimation is not explicitly optimized during the learning process. Recently, to address training inefficiencies and scalability issues, Deep Learning (DL) optimizers and architectures have been increasingly integrated with various types of FLSs [5,13,15,16]. However, most DL-based fuzzy learning frameworks mainly target point-wise prediction accuracy and do not explicitly model uncertainty, which limits their overall suitability for uncertainty-aware regression tasks.

1.1 Purpose of Thesis

The purpose of this thesis is to develop a unified, scalable, and computationally efficient learning framework for T1-FLSs and IT2-FLSs by embedding them into modern DL environments. The thesis addresses the fundamental challenges highlighted in the preceding sections, namely the high training and inference complexity of FLSs, the curse of dimensionality, and the effective utilization of uncertainty information in learning tasks.

First, the thesis focuses on enabling efficient training of T1-FLSs and IT2-FLSs on large datasets by reformulating FLS learning within DL frameworks. To this end, parameterization strategies are introduced to transform the inherently constrained learning problem of FLSs into an unconstrained form, allowing the direct use of mini batch based DL optimizers and automatic differentiation. In addition, computationally efficient and fully parallelized inference implementations are developed, including a non iterative IT2-FLS inference scheme that eliminates the iterative nature of the KMA and leverages GPU parallelization to significantly reduce training time without degrading accuracy.

Second, the thesis aims to enhance the learning capabilities of IT2-FLSs with a particular focus on uncertainty quantification through PIs. Novel CSCMs and High Dimesional Takagi-Sugeno-Kang (TSK) structures are proposed to increase design flexibility, reduce inference complexity, and mitigate the curse of dimensionality. These enhancements are designed to support dual-focused learning, where both point-wise prediction accuracy and high-quality PIs are jointly optimized.

By combining efficient DL-based training mechanisms with methodological enhancements for uncertainty aware learning, this thesis seeks to establish IT2-FLSs as scalable, practical, and reliable models for large scale and high dimensional regression problems where both accuracy and uncertainty quantification are essential.

1.2 Literature Review

Existing studies on IT2-FLSs highlight their potential for uncertainty aware regression, particularly through the generation of PIs [1,6]–[8]. However, learning IT2-FLSs remains considerably more challenging than learning T1-FLSs due to the increased number of parameters, the use of CSCMs, and the constraints required to maintain valid fuzzy set structures [5].

Most IT2-FLS learning approaches are direct extensions of T1-FLS methodologies [3,14,17]–[21]. These methods primarily focus on point wise prediction accuracy, while uncertainty related parameters are treated as additional learnable variables. As a result, critical aspects such as inference efficiency, scalability, and effective utilization of uncertainty information are often insufficiently addressed.

Recent advances integrating deep learning with fuzzy systems have demonstrated that DL based T1-FLS variants can mitigate the curse of dimensionality [10,12,15,22]. Nevertheless, most DL based fuzzy learning frameworks primarily target point wise accuracy and do not explicitly incorporate uncertainty modeling. Only a limited number of recent studies have proposed DL based IT2-FLS frameworks that exploit the Type-Reduced Set (TRS) to enable PI learning via established CSCMs [5]. However, widely adopted CSCMs such as the KMA remain computationally expensive due to their iterative and sorting based nature, restricting their applicability in large scale and parallel learning scenarios [9]. These limitations underscore the need for learning frameworks that jointly address uncertainty quantification, inference efficiency, and scalability.

1.3 Hypothesis

This thesis is based on the hypothesis that T1-FLSs and IT2-FLSs can be transformed into scalable and efficient learning models by reformulating their inference and learning mechanisms within modern DL frameworks. Specifically, it is hypothesized that:

- embedding IT2-FLSs into DL environments through appropriate parameterization can eliminate training constraints and enable the effective use of unconstrained, mini-batched DL optimizers;
- redesigning CSCMs and high dimensional TSK structures can significantly reduce inference complexity while enhancing uncertainty quantification performance;
- reformulating IT2-FLS inference in a fully vectorized and parallelizable manner can remove the iterative bottlenecks of conventional CSCMs and enable efficient execution on GPU architectures;
- exploiting GPU-based parallel computation within DL frameworks can substantially reduce training and inference time for FLSs without degrading prediction accuracy or uncertainty quality;
- a dual-focused learning strategy that jointly optimizes point-wise accuracy and prediction interval quality can outperform traditional IT2-FLS learning approaches, especially in high-dimensional and large-scale settings.

Under these hypotheses, the thesis aims to demonstrate that IT2-FLSs can achieve both computational efficiency and high-quality uncertainty-aware predictions while preserving their interpretable rule-based structure.

2. BACKGROUND ON FLSs

2.1 Overview of Fuzzy Logic Systems

A FLS is a computational framework designed to model reasoning under uncertainty, vagueness, and imprecision. Instead of relying on strict binary evaluations, fuzzy logic represents information using degrees of membership that continuously range between 0 and 1. This structure enables the system to capture gradual transitions between linguistic concepts.

FLSs operate by translating numerical inputs into linguistic terms, processing them through a set of rules, and generating an interpretable output. This approach makes fuzzy logic especially useful in situations where precise mathematical models are difficult to obtain or where expert knowledge is naturally expressed in verbal form.

However, this thesis uses FLSs as rule-based nonlinear input-output models.

2.2 Main Components of a FLS

A Fuzzy Logic System is built around a sequence of interconnected components that transform crisp numerical inputs into a final crisp output through fuzzy reasoning. Although different FLS architectures may vary in their internal structure, the fundamental workflow generally consists of four stages: fuzzification, rule evaluation, inference, and defuzzification.

2.2.1 Fuzzification

Fuzzification is the first stage in the processing pipeline, where crisp input values are mapped to fuzzy sets defined in the antecedent space. This is achieved through membership functions that assign a degree of membership to each input, reflecting how strongly the input corresponds to linguistic categories.

2.2.2 Rule base

The rule base contains the collection of fuzzy if-then rules that describe the relationship between the system inputs and outputs. These rules encode expert knowledge or learned behavior using linguistic terms, allowing the system to combine multiple antecedent conditions into a unified conclusion. Each rule contributes partially to the final output, with its influence determined by its membership value.

2.2.3 Inference mechanism

The inference mechanism determines how different rules are activated and combined. After the antecedent membership values are obtained in the fuzzification stage, the inference process computes a firing strength for each rule, representing how applicable the rule is under the given input. These rule activations are then aggregated to produce a combined fuzzy output. Different FLS types (e.g., Mamdani or TSK) employ different inference strategies, but the overall role of the inference mechanism remains the same: integrating the contributions of all rules into a coherent fuzzy decision.

2.2.4 Defuzzification

Defuzzification is the final stage, in which the aggregated fuzzy output is transformed back into a crisp numerical value suitable for use in control, estimation, or decision-making tasks. In the case of Mamdani-type systems, defuzzification typically involves computing a centroid or weighted average from a fuzzy output set. In contrast, TSK-type systems generate rule outputs in functional form, allowing the final crisp output to be obtained directly from a weighted combination of these functions.

2.3 Membership Functions

A MF is a mathematical function that defines how much a crisp input value belongs to a fuzzy set. It assigns each input a value between 0 and 1, where values close to 1 indicate a high degree of belonging and values close to 0 indicate a low degree of belonging. Membership functions are used to represent gradual transitions between linguistic concepts instead of sharp boundaries.

2.3.1 Antecedent membership functions

The antecedent part of each fuzzy rule is defined by membership functions that describe how strongly an input belongs to a certain group or linguistic category. These functions determine the degree of activation of each rule and the influence on the behavior of the system. Several shapes can be used depending on the desired smoothness, interpretability, and modeling flexibility.

Commonly used membership function types include triangular, trapezoidal, Gaussian, bell-shaped, and sigmoidal forms. In Type-2 fuzzy systems, each membership function is extended to include a footprint of uncertainty, allowing the grade of membership to vary within an interval.

In this thesis, Gaussian membership functions are used for the antecedent part of each rule.

2.3.2 Consequent membership functions

The consequent part of a fuzzy rule defines the output associated with each antecedent condition. Different fuzzy system types use different structures for the consequent, which significantly affects the computational complexity and the interpretability of the system.

In Mamdani-type fuzzy systems, the consequent is expressed as a fuzzy set, meaning that each rule generates a fuzzy output that must later be defuzzified. This structure is intuitive and aligns closely with human reasoning but may require more computations.

TSK-type fuzzy systems define each consequent as either a constant value or a function of the inputs, often linear. Because the output of each rule is already crisp, the final system output can be obtained directly through a weighted average of the rule consequents.

In this thesis, linear consequent functions are used.

2.4 Categories of FLSs

Fuzzy Logic Systems can be categorized according to the structure of their rule consequents and the way uncertainty is represented. In terms of the rule consequent, the two main architectures are Mamdani and TSK FLSs. In Mamdani FLSs, both antecedents and consequents are defined by fuzzy sets, and a defuzzification step is required to obtain a crisp output. In contrast, TSK FLSs define the consequent of each rule as a constant or, more commonly, as a linear function of the inputs, which allows the system output to be computed directly as a weighted combination of the rule consequents. From the perspective of uncertainty representation, FLSs can be further classified as Type-1 or Type-2 systems. Type-1 FLSs employ precise membership functions with crisp membership grades, whereas Type-2 FLSs introduce uncertainty into the membership grades through a footprint of uncertainty, leading to improved robustness at the cost of increased computational complexity. In this thesis, only TSK FLSs are considered, including T1 and IT2 formulations.

2.5 Mathematical Definition of FLS

2.5.1 Type-1 FLS

A T1-FLS consists of P fuzzy rules $(p = 1, 2, \ldots, P)$ defined for an input vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_M)^T$ and an output variable y. Each rule in the system defines a fuzzy relationship between the inputs and the output using Type-1 fuzzy sets. The general rule structure can be expressed as:

$$R_p : \text{If } x_1 \text{ is } A_{p,1} \text{ and} \ldots x_M \text{ is } A_{p,M} \text{ Then } y \text{ is } y_p \tag{2.1}$$

where $A_{p,m}$ represents the fuzzy set corresponding to the m^{th} input in the p^{th} rule, and y_p is the rule output, typically modeled as a linear function of the inputs:

$$y_p = \sum_{m=1}^{M} a_{p,m} x_m + a_{p,0} \tag{2.2}$$

The overall output of the T1-FLS is obtained through a weighted average of the rule outputs, where the weights correspond to the firing strengths of each rule:

$$y_{T1}(\boldsymbol{x}) = \frac{\sum_{p=1}^{P} f_p(\boldsymbol{x}) y_p}{\sum_{p=1}^{P} f_p(\boldsymbol{x})} \tag{2.3}$$

Here, $f_p = \prod_{m=1}^{M} \mu_{A_{p,m}}$ denotes the firing strength (or activation level) of the p^{th} rule, determined by the product of the membership grades of its antecedents. The membership function (MF) of each antecedent fuzzy set $A_{p,m}$ is commonly defined as a Gaussian function illustrated in Figure 2.1:

$$\mu_{A_{p,m}}(x_m) = \exp\left(-\frac{(x_m - c_{p,m})^2}{2\sigma_{p,m}^2}\right) \tag{2.4}$$

In this expression, $c_{p,m}$ and $\sigma_{p,m}$ represent the center and the standard deviation of the Gaussian membership function, respectively. This formulation allows the T1-FLS to map nonlinear input–output relationships smoothly by blending the contributions of all rules according to their activation strengths.

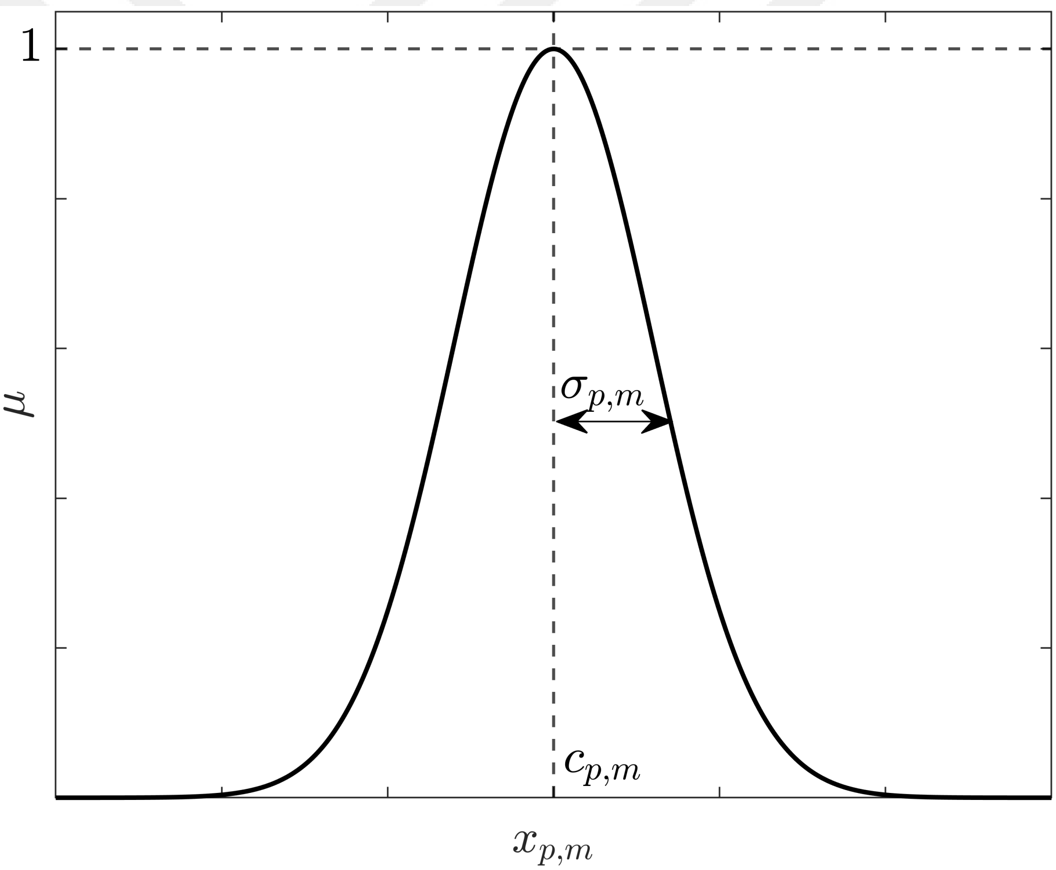


Figure 2.1: Gaussian T1-MF.

2.5.2 Interval Type-2 FLS

An IT2-FLS consists of a set of P fuzzy rules $(p = 1,2,\ldots,P)$ defined for an input vector $x = (x_1,x_2,\ldots,x_M)^T$ and an output variable y. Each rule in the system describes the relationship between the input and output variables using IT2-FSs. The general structure of an IT2-FLS rule can be expressed as:

$$R_p : \text{If } x_1 \text{ is } \tilde{A}_{p,1} \text{ and} \ldots x_M \text{ is } \tilde{A}_{p,M} \text{ Then } y \text{ is } y_p \tag{2.5}$$

where $\tilde{A}_{p,m}$ denotes the interval type-2 fuzzy set associated with the m^{th} input in the p^{th} rule, and y_p represents the rule output, which is typically defined as a linear function of the inputs:

$$y_p = \sum_{m=1}^{M} a_{p,m}x_m + a_{p,0} \tag{2.6}$$

Here, $a_{p,m}$ and $a_{p,0}$ are the consequent parameters of the p^{th} rule. This rule structure enables the IT2-FLS to handle uncertainty more effectively than a Type-1 FLS by incorporating a footprint of uncertainty in the membership functions of the antecedent fuzzy sets illustrated in Figure 2.2.

The output of IT2-FLS is defined as:

$$y_{IT2}(\boldsymbol{x}) = (\underline{y}_{IT2}(\boldsymbol{x}) + \overline{y}_{IT2}(\boldsymbol{x}))/2 \tag{2.7}$$

and $\underline{y}_{IT2}$ and $\overline{y}_{IT2}$, bounds of the type reduced set, are obtained by optimizing:

$$\begin{aligned}
\underline{y}_{IT2}(\boldsymbol{x}) &= \min_{L\in[1,P-1]} \frac{\sum_{p=1}^{L}\overline{f}_p(\boldsymbol{x})y_p + \sum_{p=L+1}^{P}\underline{f}_p(\boldsymbol{x})y_p}{\sum_{p=1}^{L}\overline{f}_p(\boldsymbol{x}) + \sum_{p=L+1}^{P}\underline{f}_p(\boldsymbol{x})} \\
\overline{y}_{IT2}(\boldsymbol{x}) &= \max_{R\in[1,P-1]} \frac{\sum_{p=1}^{R}\underline{f}_p(\boldsymbol{x})y_p + \sum_{p=R+1}^{P}\overline{f}_p(\boldsymbol{x})y_p}{\sum_{p=1}^{R}\underline{f}_p(\boldsymbol{x}) + \sum_{p=R+1}^{P}\overline{f}_p(\boldsymbol{x})}
\end{aligned} \tag{2.8}$$

where L,R are the switching points of the KMA [23]. $\underline{f}_p$ and $\overline{f}_p$ are the lower and upper rule firing of the p^{th} rule and are defined as:

$$\underline{f}_p = \prod_{m=1}^{M} \underline{\mu}_{\tilde{A}_{p,m}}, \overline{f}_p = \prod_{m=1}^{M} \overline{\mu}_{\tilde{A}_{p,m}} \tag{2.9}$$

Here, $\overline{\mu}_{\tilde{A}_{p,m}}$ and $\underline{\mu}_{\tilde{A}_{p,m}}$ are the UMFs and LMFs, respectively:

$$\begin{aligned} \overline{\mu}_{\tilde{A}_{p,m}}(x_m) &= \exp\left(-\left(x_m - c_{p,m}\right)^2 / 2\overline{\sigma}_{p,m}^2\right) \\ \underline{\mu}_{\tilde{A}_{p,m}}(x_m) &= h_{p,m} \exp\left(-\left(x_m - c_{p,m}\right)^2 / 2\underline{\sigma}_{p,m}^2\right) \end{aligned} \tag{2.10}$$

where $c_{p,m}$ is the center, $\tilde{\sigma}_{p,m} = [\underline{\sigma}_{p,m}, \overline{\sigma}_{p,m}]$ is the standard deviation while $h_{p,m}$ defines the height of the LMF $\forall p, m$.

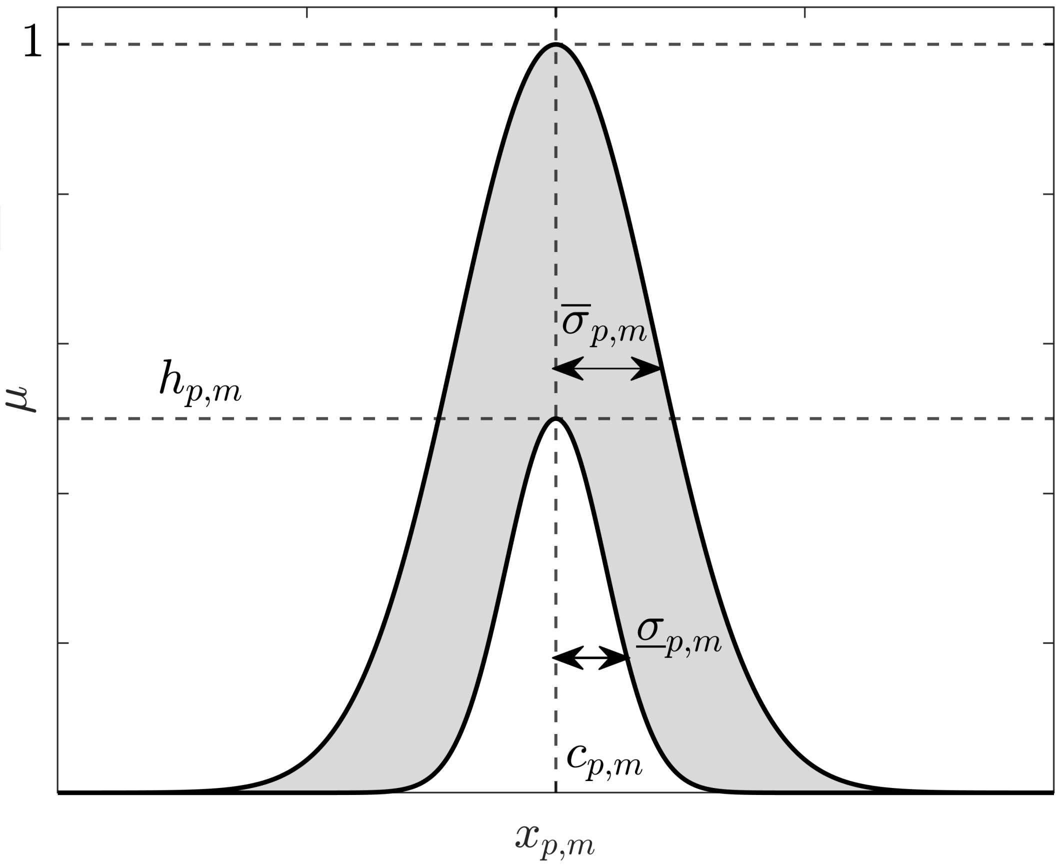


Figure 2.2: Gaussian IT2-MF.

3. TRAINING FLSs WITHIN DL FRAMEWORKS

3.1 Purpose

The purpose of this chapter is to present a unified deep learning framework which proposed in [24] to train T1 and IT2 FLSs. Traditional learning methods become inefficient when the number of samples, input dimensions, or rule count increases [11], [12], mainly due to the constrained parameters and the computational cost of inference [5], [15]. To overcome these limitations, we reformulate the FLS components using matrix-based and mini-batch-oriented expressions, allowing the use of modern DL optimizers and automatic differentiation. This enables the training of T1 and IT2-FLSs in the same manner as neural networks while preserving their interpretable, rule-based structure.

The framework integrates learnable antecedent, consequent, and CSCM parameters, supports GPU execution through vectorized operations, and provides scalable inference for both T1 and IT2 systems. This chapter therefore establishes the theoretical basis and computational tools necessary for efficient large-scale learning of fuzzy systems.

3.2 Handling the Constraints and Parameterization Tricks

Training T1 and IT2-FLSs requires satisfying the structural constraints that come from the definitions of their membership functions and fuzzy sets. However, deep learning optimizers operate in an unconstrained parameter space and cannot directly enforce bounds, positivity conditions, or ordering relations. Therefore, parameterization tricks are needed to convert all constrained variables into unconstrained ones. By representing each constrained parameter through an unconstrained auxiliary variable, the entire FLS can be trained with standard DL optimizers while still ensuring that all fuzzy set definitions remain valid.

3.2.1 Learnable parameter sets

We define and group the LPs of the T1-FLSs and IT2-FLSs (θ) regarding whether they are defined within the antecedents (θ_A), consequents (θ_C) or CSCM (θ_M) parts.

- For T1-FLS, θ_A is defined as $\theta_A = \{c, \sigma\}$, where $c = (c_{1,1}, \ldots, c_{P,M})^T \in \mathbb{R}^{P\times M}$, $\sigma = (\sigma_{1,1}, \ldots, \sigma_{P,M})^T \in \mathbb{R}^{P\times M}$. θ_A has in total $2PM$ LPs.
- For IT2-FLS, θ_A is defined as $\theta_A = \{c, \underline{\sigma}, \overline{\sigma}, h\}$, where $c = (c_{1,1}, \ldots, c_{P,M})^T \in \mathbb{R}^{P\times M}$, $\underline{\sigma} = (\underline{\sigma}_{1,1}, \ldots, \underline{\sigma}_{P,M})^T \in \mathbb{R}^{P\times M}$, $\overline{\sigma} = (\overline{\sigma}_{1,1}, \ldots, \overline{\sigma}_{P,M})^T \in \mathbb{R}^{P\times M}$, $h = (h_{1,1}, \ldots, h_{P,M})^T \in \mathbb{R}^{P\times M}$. θ_A has in total $4PM$ LPs.
- $\theta_C = \{a, a_0\}$, with $a = (a_{1,1}, \ldots, a_{P,M})^T \in \mathbb{R}^{P\times M}$ and $a_0 = (a_{1,0}, \ldots, a_{P,0})^T \in \mathbb{R}^{P\times 1}$. θ_C has in total $P(M+1)$ LPs.
- θ_M depends on the deployed CSCMs in IT2-FLS and is not used in T1-FLS.

The total number of learnable parameters in an IT2-FLS depends on the deployed CSCM, since CSCM specific parameters may introduce addition antecedent and consequent LPs. To make this explicit, we define θ for the CSCMs considered in this thesis, including both conventional methods and the CSCMs proposed in Sec. 4.2: Karnik-Mendel (KM), Simple Method (SM), Nie-Tan (NT), Weighted Karnik-Mendel (WKM), Begian-Melek-Mendel (BMM), Weighted Nie-Tan (WNT), Multi Weighted Nie-Tan (MWNT), and Parametric Karnik-Mendel (PKM). We define θ for an IT2-FLS deploying:

- KM/SM/NT: $\theta = \{\theta_A, \theta_C\}$. Thus, #LP is $5PM + P$.
- WKM/BMM/WNT: $\theta = \{\theta_A, \theta_C, \theta_M\}$. #LP is $5PM + P + 1$, where $\theta_M = \beta$ and $\beta \in \mathbb{R}$.
- MWNT: $\theta = \{\theta_A, \theta_C, \theta_M\}$. #LP is $5PM + 2P$ with $\theta_M = \gamma$ and $\gamma = (\gamma_1, \ldots, \gamma_P)^T \in \mathbb{R}^{P\times 1}$
- PKM: $\theta = \{\theta_A, \theta_C, \theta_M\}$. #LP is $5PM + 3P$ where $\theta_M = \{\underline{v}, \overline{v}\}$ and $\underline{v} = (\underline{v}_1, \ldots, \underline{v}_P)^T \in \mathbb{R}^{P\times 1}$, $\overline{v} = (\overline{v}_1, \ldots, \overline{v}_P)^T \in \mathbb{R}^{P\times 1}$.

3.2.2 Parameterization tricks for DL Optimizers

The learning problem of both T1-FLS and IT2-FLS are defined with constraints $\theta \in C$, arising from the FS and FLS definitions. Given that DL optimizers are unconstrained techniques, we present tricks to transform θ to an unbounded search space.

For the LPs of T1-FLS, $\sigma_{p,m} \in [0, \infty]$ we employ the following trick:

$$\sigma_{p,m} = abs(\sigma'_{p,m}) \tag{3.1}$$

where $\sigma'_{p,m} \in [-\infty, \infty], \forall p, m$ are the new and unbounded optimization variables.

For the LPs of IT2-FLS, $h_{p,m} \in [0, 1]$ and $\{\underline{\sigma}_{p,m}, \overline{\sigma}_{p,m}\}$ ($\underline{\sigma}_{p,m} \leq \overline{\sigma}_{p,m}$), we employ the following tricks given in [5]:

$$\begin{gathered} h_{p,m} = sigmoid(h'_{p,m}) \\ \underline{\sigma}_{p,m} = \sigma'_{p,m} - \left|\Delta_{p,m}\right| \quad \overline{\sigma}_{p,m} = \sigma'_{p,m} + \left|\Delta_{p,m}\right| \end{gathered} \tag{3.2}$$

where $\{h'_{p,m}, \sigma'_{p,m}, \Delta_{p,m}\} \in [-\infty, \infty], \forall p, m$ are the new and unbounded optimization variables.

Also for CSCM of IT2-FLSs we employ the following tricks given in [25]:

- For the WKM/WNT/BMM, we must satisfy $\beta \in [0, 1]$. In this context, we define a new LP $\beta' \in [-\infty, \infty]$ as:

$$\beta = sigmoid\left(\beta'\right) \tag{3.3}$$

- For the MWNT, we have to satisfy $\gamma_p \in [0, 1], \forall p$. Thus, we define new LPs $\gamma'_p \in [-\infty, \infty], \forall p$ as follows:

$$\gamma_p = sigmoid\left(\gamma'_p\right) \tag{3.4}$$

- For the PKM, we have a more strict constraint as $\{\underline{\nu}_p, \overline{\nu}_p\} \in \{0, 1\}, \forall p$. We relax this constraint by defining new LPs $\{\underline{\nu}'_p, \overline{\nu}'_p\} \in [-\infty, \infty], \forall p$ as follows:

$$\underline{\nu}_p = sigmoid\left(10\underline{\nu}'_p\right), \overline{\nu}_p = sigmoid\left(10\overline{\nu}'_p\right) \tag{3.5}$$

  The scaling of 10 increases the $sigmoid(.)$'s slope, aiming to approximate the $sign(.)$ function while being differentiable. Thus, LPs almost satisfy $(\{\underline{\nu}_p, \overline{\nu}_p\} \in \{0, 1\})$.

3.3 Scaling T1-FLS Inference to Mini-batches

We developed an efficient T1-FLS inference implementation for a mini-batch $S^{(i)}$ in [24]. For a given θ, we first define $\{c^{Rep}, \sigma^{Rep}\} \in \mathbb{R}^{P\times M\times B}$ and $a_0^{Rep} \in \mathbb{R}^{P\times B}$:

$$\{c^{Rep}, \sigma^{Rep}\} = \{c, \sigma\} \otimes \mathbb{I}^{1\times 1\times B} \,,\, a_0^{Rep} = a_0 \otimes \mathbb{I}^{1\times B} \tag{3.6}$$

and then the inference is accomplished with the following steps:

- Calculate $\mu_A \in \mathbb{R}^{P\times M\times B}$ for a $S^{(i)}$ with:

$$\mu_A = \exp\left((x'^{Rep} \ominus c^{Rep})^2 \oslash 2(\sigma^{Rep})^2\right) \tag{3.7}$$

by computing $x'^{Rep} \in \mathbb{R}^{P\times M\times B}$ from:

$$x'^{Rep} = x' \otimes \mathbb{I}^{P\times 1\times 1} \tag{3.8}$$

with $x' \in \mathbb{R}^{1\times M\times B}$

$$x' = Permute(x^{(i)}_{1:B}) \text{ from } (0,1,2) \text{ to } (2,0,1) \tag{3.9}$$

- Calculate $f_p, \forall p, m, n$:

$$f = \prod_{m=1}^{M} \mu_A \,,\, f \in \mathbb{R}^{P\times 1\times B} \tag{3.10}$$

- The output $y_{T1} \in \mathbb{R}^{1\times 1\times B}$ for a $S^{(i)}$ is calculated via:

$$y_{T1} = \sum_P (f_{norm} \odot y'_p) \tag{3.11}$$

where $f_{norm} \in \mathbb{R}^{P\times 1\times B}$ is obtained through:

$$\begin{gathered} f_{norm} = f \oslash f_{sum}^{Rep} \\ f_{sum}^{Rep} = f_{sum} \otimes \mathbb{I}^{P\times 1\times 1} \,,\, f_{sum}^{Rep} \in \mathbb{R}^{P\times 1\times B} \\ f_{sum} = \sum_P f \,,\, f_{sum} \in \mathbb{R}^{1\times 1\times B} \end{gathered} \tag{3.12}$$

and $y'_p \in \mathbb{R}^{P\times 1\times B}$ derived by first computing

$$y_p = a\boldsymbol{x}^{(i)}_{1:B} \oplus a_0^{Rep} \,,\, y_p \in \mathbb{R}^{P\times B} \tag{3.13}$$

followed by the permutation:

$$y'_p = Permute(y_p) \text{ from } (0,1,2) \text{ to } (0,2,1). \tag{3.14}$$

3.4 Scaling IT2-FLS Inference to Mini-batches

The inference time and complexity of the IT2-FLS is high due to the iterative nature of KM Algorithm which also requires a sorting procedure. It has been shown in [26] that (2.8) can be reformulated via $u_p \in 0,1$ which defines an equivalent $f_p = \overline{f}_p u_p + \underline{f}_p(1-u_p)$. Yet, this new formulation creates P new optimization variables and still requires finding $\underline{y}_{IT2}$ and $\overline{y}_{IT2}$ iteratively.

In [24], we presented an efficient IT2-FLS inference implementation by eliminating the required optimization problems in (2.8) by evaluating

$$Y(\boldsymbol{u}) = X(\boldsymbol{u}) \oslash Z(\boldsymbol{u}) \tag{3.15}$$

$$\begin{aligned} X(\boldsymbol{u}) &= \alpha_0 + \alpha\boldsymbol{u}: \quad \alpha_0 = \textstyle\sum_p^P y_p \underline{f}_p \ ; \ \alpha_p = y_p(\overline{f}_p - \underline{f}_p) \ , \ p = 1, \dots, P \\ Z(\boldsymbol{u}) &= \beta_0 + \beta\boldsymbol{u}: \quad \beta_0 = \textstyle\sum_p^P \underline{f}_p \ ; \ \beta_p = (\overline{f}_p - \underline{f}_p) \ , \ p = 1, \dots, P \end{aligned} \tag{3.16}$$

with $u \in \mathbb{R}^{P \times 2^P}$ that defines all binary combinations u_p as follows:

$$\boldsymbol{u} = \prod_{p=1}^{P} \{0,1\}. \tag{3.17}$$

Here, $\prod$ denotes the cartesian product. For instance, u for $P = 3$ is as follows:

$$\boldsymbol{u} = \prod_{p=1}^{3} \{0,1\} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \tag{3.18}$$

As $Y(\boldsymbol{u})$ includes all possible solutions, we can obtain seamlessly $\underline{y}_{IT2}$ and $\overline{y}_{IT2}$:

$$\underline{y}_{IT2} = \min(Y(\boldsymbol{u})) \ , \ \overline{y}_{IT2} = \max(Y(\boldsymbol{u})) \tag{3.19}$$

Note that, we chose this inference implementation because GPUs are renowned for their efficiency in executing matrix operations.

Let us now extend the IT2-FLS inference to a mini-batch. For a given θ, we define the variables $\{c^{Rep}, h^{Rep}\} \in \mathbb{R}^{P \times M \times B}$ and $\{\underline{\sigma}^{Rep}, \overline{\sigma}^{Rep}\} \in \mathbb{R}^{P \times M \times B}$:

$$\{c^{Rep}, h^{Rep}\} = \{c, h\} \otimes \mathbb{I}^{1 \times 1 \times B} \ , \ \{\underline{\sigma}^{Rep}, \overline{\sigma}^{Rep}\} = \{\underline{\sigma}, \overline{\sigma}\} \otimes \mathbb{I}^{1 \times 1 \times B} \tag{3.20}$$

and then the inference is accomplished with the following steps:

- Calculate $\overline{\mu}_{\tilde{A}} \in \mathbb{R}^{P\times M\times B}$ and $\underline{\mu}_{\tilde{A}} \in \mathbb{R}^{P\times M\times B}$ as follows:

$$\begin{aligned}\overline{\mu}_{\tilde{A}} &= \exp\left((x'^{Rep} \ominus c^{Rep})^2 \oslash 2(\overline{\sigma}^{Rep})^2\right)\\ \underline{\mu}_{\tilde{A}} &= h^{Rep} \otimes \exp\left((x'^{Rep} \ominus c^{Rep})^2 \oslash 2(\underline{\sigma}^{Rep})^2\right)\end{aligned} \tag{3.21}$$

by performing the following operations on the input:

$$x' = Permute(x^{(i)}_{1:B}) \text{ from } (0,1,2) \text{ to } (2,0,1)\ ,\ x' \in \mathbb{R}^{1\times M\times B} \tag{3.22}$$

$$x'^{Rep} = x' \otimes \mathbb{I}^{P\times 1\times 1}\ ,\ x'^{Rep} \in \mathbb{R}^{P\times M\times B} \tag{3.23}$$

- Calculate $\{\underline{f}, \overline{f}\} \in \mathbb{R}^{P\times 1\times B}$ for the mini-batch:

$$\underline{f} = \prod_{m=1}^{M} \underline{\mu}_{\tilde{A}}\ ,\ \overline{f} = \prod_{m=1}^{M} \overline{\mu}_{\tilde{A}} \tag{3.24}$$

- Compute the component $Z(u) \in \mathbb{R}^{1\times 2^P\times B}$ of (3.15):

$$\boldsymbol{Z}(u) = \boldsymbol{\beta}_0^{Rep} \oplus \boldsymbol{\beta}' \tag{3.25}$$

by evaluating $\beta_0^{Rep} \in \mathbb{R}^{1\times 2^P\times B}$ via:

$$\boldsymbol{\beta}_0^{Rep} = \boldsymbol{\beta}_0 \otimes \mathbb{I}^{1\times 2^P\times 1}\ ,\ \boldsymbol{\beta}_0 = \sum_{p=1}^{P} \underline{f}\ ,\ \boldsymbol{\beta}_0 \in \mathbb{R}^{1\times 1\times B} \tag{3.26}$$

and $\beta' \in \mathbb{R}^{1\times 2^P\times B}$ via:

$$\boldsymbol{\beta}' = Permute(\boldsymbol{\beta}) \text{ from } (0,1,2) \text{ to } (2,1,0)\ ,\ \boldsymbol{\beta} = \Delta f' u\ ,\ \boldsymbol{\beta} \in \mathbb{R}^{B\times 2^P} \tag{3.27}$$

where $\Delta f' \in \mathbb{R}^{B\times P}$ is derived from

$$\Delta f = \overline{f} \ominus \underline{f}\ ,\ \Delta f \in \mathbb{R}^{P\times 1\times B} \tag{3.28}$$

through the following permutation

$$\Delta f' = Permute(\Delta f) \text{ from } (0,1,2) \text{ to } (2,0,1). \tag{3.29}$$

- Calculate the component $X(u) \in \mathbb{R}^{1\times 2^P\times B}$ of (3.15):

$$\boldsymbol{X}(u) = \boldsymbol{\alpha}_0^{Rep} \oplus \boldsymbol{\alpha}'. \tag{3.30}$$

Here, $\alpha' \in \mathbb{R}^{1\times 2^P \times B}$ is obtained from $\alpha^{Tp} \in \mathbb{R}^{B\times P}$ as follows:

$$\boldsymbol{\alpha}' = Permute(\boldsymbol{\alpha}) \text{ from } (0,1,2) \text{ to } (2,1,0) \tag{3.31}$$

$$\boldsymbol{\alpha} = \boldsymbol{\alpha}^{Tp} u \ , \ \boldsymbol{\alpha} \in \mathbb{R}^{B\times 2^P} \ , \ \boldsymbol{\alpha}^{Tp} = (y''_p \odot \Delta f') \tag{3.32}$$

where $y''_p \in \mathbb{R}^{B\times P}$ is obtained from y_p (given in (3.13)) by performing:

$$y''_p = Permute(y_p) \text{ from } (0,1,2) \text{ to } (2,0,1) \tag{3.33}$$

and $\alpha_0^{Rep} \in \mathbb{R}^{1\times 2^P \times B}$ is computed using y'_p (given in (3.14)) through:

$$\boldsymbol{\alpha}_0^{Rep} = \boldsymbol{\alpha}_0 \otimes \mathbb{I}^{1\times 2^P \times 1} \ , \ \boldsymbol{\alpha}_0 = \textstyle\sum_P (y'_p \odot \underline{f}) \ , \ \boldsymbol{\alpha}_0 \in \mathbb{R}^{1\times 1\times B} \tag{3.34}$$

- Compute $Y(u) \in \mathbb{R}^{1\times 2^P \times B}$

$$\boldsymbol{Y}(u) = \boldsymbol{X}(u) \oslash \boldsymbol{Z}(u) \tag{3.35}$$

and obtain $\underline{y}_{IT2} \in \mathbb{R}^{1\times 1\times B}$ and $\overline{y}_{IT2} \in \mathbb{R}^{1\times 1\times B}$ simply with

$$\underline{y}_{IT2} = \min(\boldsymbol{Y}(u)(:,:,b)) \ , \ \overline{y}_{IT2} = \max(\boldsymbol{Y}(u)(:,:,b)) \ , \forall b \tag{3.36}$$

to calculate $y_{IT2} \in \mathbb{R}^{1\times 1\times B}$:

$$y_{IT2} = (\underline{y}_{IT2} \oplus \overline{y}_{IT2})/2 \tag{3.37}$$

Remark: Note that while the FLS inference is demonstrated for a single output, the implementation can be extended to accommodate multi-output FLSs. The multi-output version of this implementation succesfully used in [25].

3.5 Learning Dual-Focused IT2-FLS

Here, we introduce a DL framework to learn an IT2-FLS capable of yielding accurate predictions and generating HQ-PI. Algorithm 1 provides the pseudo code for training IT2-FLS on a dataset $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$, where $\boldsymbol{x}_n = (x_{n,1}, \ldots, x_{n,M})^T$.

To train a dual-focused IT2-FLS, we define a composite loss consisting of an accuracy-focused $L_R(\cdot)$ and an uncertainty-focused part $\ell(\cdot)$ [5], as:

$$\min_{\boldsymbol{\theta}\in\boldsymbol{C}} L = \frac{1}{N} \textstyle\sum_{n=1}^{N} \left[L_R(\boldsymbol{x}_n, y_n) + \ell(\boldsymbol{x}_n, y_n, \underline{\tau}, \overline{\tau})\right] \tag{3.38}$$

The constraints ($\boldsymbol{C}$) are eliminated as described in Section 3.2.2.

For the accuracy-focused part, we define $L_R(.)$ with:

$$L_R(\varepsilon_n) = \log\left(\cosh\left(\varepsilon_n\right)\right) \tag{3.39}$$

where $\varepsilon_n = y_n - y\left(x_n\right)$. For uncertainty coverage purposes, we construct $\ell(.)$ based on two tilted loss functions as follows:

$$\ell\left(\boldsymbol{x}_n, y_n, \underline{\tau}, \overline{\tau}\right) = \underline{\ell}\left(\boldsymbol{x}_n, y_n, \underline{\tau}\right) + \overline{\ell}\left(\boldsymbol{x}_n, y_n, \overline{\tau}\right) \tag{3.40}$$

with

$$\begin{aligned} \underline{\ell} &= \max(\underline{\tau}(y_n - \underline{y}(\boldsymbol{x}_n)), (\underline{\tau} - 1)(y_n - \underline{y}(\boldsymbol{x}_n))) \\ \overline{\ell} &= \max(\overline{\tau}(y_n - \overline{y}(\boldsymbol{x}_n)), (\overline{\tau} - 1)(y_n - \overline{y}(\boldsymbol{x}_n))) \end{aligned} \tag{3.41}$$

Here, $\underline{\tau}$ represents a lower quantile while $\overline{\tau}$ an upper quantile to learn an expected amount of uncertainty, i.e. $\varphi = [\underline{\tau}, \overline{\tau}]$.

To sum up, the final composite loss is defined as follows:

$$L = \frac{1}{N}\sum_{n=1}^{N}\left[L_R\left(\varepsilon_n\right) + \ell\left(x_n, y_n, \underline{\tau}, \overline{\tau}\right)\right] \tag{3.42}$$

Thus, for instance, we can aim to learn an IT2-FLS that ensures a PI of $I^{\varphi} = 99\%$ (by setting $\varphi = [0.005, 0.995]$) via the TRS $[\underline{y}(x_n), \overline{y}(x_n)]$, while minimizing ε_n (i.e accuracy) via the output $y(x_n)$ of the IT2-FLS.

3.6 GPU Integration

The matrix and mini-batch-based formulations of the T1-FLS and IT2-FLS developed in the Sec. 3.3 and 3.4 naturally enable efficient execution on GPUs. Since the rule firing strengths, CSCMs, and defuzzification steps are expressed using element-wise operations, matrix multiplications, and reductions over the rule and batch dimensions, no algorithmic modification is required for GPU integration. Instead, the same equations are evaluated on data structures residing in GPU memory. In practice, the input mini-batches, the learnable parameter sets θ_A, θ_C, and θ_M, and the intermediate quantities (membership grades, firing strengths, TRS bounds, and losses) are all stored as GPU-resident arrays, while the forward and backward passes follow the same computational graph as in the CPU case (cf. Sec. 3.5 and Sec. 3.2.2). In this study, this strategy was implemented using MATLAB's GPU driver support, where the mini-batch tensors and parameter matrices were cast to GPU arrays and all core operations were executed using built-in GPU-enabled linear algebra routines.

Algorithm 1 DL-based Dual-Focused IT2-FLS with eHTSK2 and PKM

Input: N training samples $(x_n, y_n)_{n=1}^{N}$, $\varphi = [\underline{\tau}, \overline{\tau}]$
P, number of rules
mbs, mini-batch size
T, number of epochs
Output: Learnable parameter set θ
Initialize $\theta = [\theta_A, \theta_C, \theta_M]$;
for $t = 1$ to T **do**
    for each *mbs* in N **do**
        Perform parametrization tricks for θ; ▷ *Sec. 3.2.2*
        Compute UMF & LMF $[\underline{\mu}, \overline{\mu}] \leftarrow \mathrm{MF}(x; \theta_A)$ ▷ *Eq.* (2.10)
        Compute firing strengths $[\underline{f}, \overline{f}] \leftarrow F(\underline{\mu}, \overline{\mu})$ ▷ *Eq.* (4.26)
        Compute TRS $[\underline{y}, \overline{y}] \leftarrow \mathrm{PKM}(\underline{f}, \overline{f}, x; [\theta_C, \theta_M])$ ▷ *Eq.(4.3,4.7)*
        $y \leftarrow \mathrm{Defuzzification}(\underline{y}, \overline{y})$ ▷ *Eq.* (4.5)
        Compute L ▷ *Eq.* (3.42)
        Compute the gradient $\partial L / \partial \theta$
        Update θ via Adam optimizer
    end for
end for
$\theta^* = \arg\min(L)$
Return $\theta = \theta^*$

4. ENHANCEMENTS TO IT2-FLS

4.1 Purpose

The purpose of this chapter is to present advanced methodological enhancements for IT2-FLSs that improve both learning efficiency and scalability in high-dimensional settings, as developed in [25] and supported by the broader analyses in [5]. In particular, the chapter focuses on two tightly connected challenges: the computational and optimization limitations of classical CSCMs used in IT2-FLSs, and the curse of dimensionality that emerges when fuzzy systems are applied to large dimensional inputs.

4.2 Enhancements to CSCMs for Learning

Classical CSCMs, such as the Karnik–Mendel algorithms, were not originally designed for gradient-based learning, parallel computation, or vectorized execution, which limits their applicability within modern deep learning frameworks. In [25], new formulations such as WKM, PKM, MWNT, and WNT were proposed to improve computational efficiency, enable differentiable and vectorized implementations, and incorporate learnable parameters for enhanced uncertainty modeling.

4.2.1 KM CSCMs for DL

The KM is known for its extensive computational needs as a sorting operation is needed and the TRS is obtained iteratively [1]. It has been shown in [26] that:

- To obtain $\underline{y}$, the KM iteratively minimizes (2.8) by finding the optimal combination of $\underline{f}$ and $\overline{f}$, namely $\underline{f}^*$. For a rule, $\underline{f}_p^*(x_p)$ is defined as follows:

$$\underline{f}_p^*(x_p) = \underline{u}_p \overline{f}_p(x_p) + (1-\underline{u}_p)\underline{f}_p(x_p) \tag{4.1}$$

  where $\underline{u}_p \in \{0,1\}$ is the binary optimization variable.

- To calculate $\overline{y}$, KM iteratively maximizes (2.8) by finding an optimal $\overline{f}_p^*(x_p), \forall p$ which is defined as follows:

$$\overline{f}_p^*(x_p) = \overline{u}_p \overline{f}_p(x_p) + (1-\overline{u}_p)\underline{f}_p(x_p) \tag{4.2}$$

  where $\overline{u}_p \in \{0,1\}, \forall p$ is the binary optimization variable.

Then, the TRS is defined as:

$$\underline{y}(\boldsymbol{x}) = \frac{\sum_{p=1}^{P} \underline{f}_p^*(x_p) y_p}{\sum_{p=1}^{P} \underline{f}_p^*(x_p)}, \overline{y}(\boldsymbol{x}) = \frac{\sum_{p=1}^{P} \overline{f}_p^*(x_p) y_p}{\sum_{p=1}^{P} \overline{f}_p^*(x_p)} \tag{4.3}$$

with

$$\begin{aligned} \underline{f}_p(\boldsymbol{x}) &= \underline{\mu}_{\tilde{A}_{p,1}} \cap \ldots \cap \underline{\mu}_{\tilde{A}_{p,M}} \\ \overline{f}_p(\boldsymbol{x}) &= \overline{\mu}_{\tilde{A}_{p,1}} \cap \ldots \cap \overline{\mu}_{\tilde{A}_{p,M}} \end{aligned} \tag{4.4}$$

where $\cap$ is the t-norm product operator. The defuzzified output is the average of $\underline{y}(\boldsymbol{x})$ and $\overline{y}(x)$ [1].

$$y(\boldsymbol{x}) = \left(\underline{y}(\boldsymbol{x}) + \overline{y}(\boldsymbol{x})\right)/2 \tag{4.5}$$

Remark 1. There does exist a direct relationship between the Left (L) and Right (R) switching points of KM [1] and the defined optimization variables $\underline{u}_p \in \{0,1\}$ and $\overline{u}_p \in \{0,1\}, \forall p$. Please see [26] for details.

Remark 2. The KM is implemented with the interface in Sec. 3.4

4.2.1.1 Weighted KM (WKM)

The proposed WKM in [25], has an extra degree of freedom by incorporating an LP, $\beta \in [0,1]$, into the defuzzification stage of KM as follows:

$$y(\boldsymbol{x}) = \beta \underline{y}(\boldsymbol{x}) + (1-\beta)\overline{y}(\boldsymbol{x}) \tag{4.6}$$

Yet, the inference complexity of WKM is the same as the KM.

4.2.1.2 Parametric KM (PKM)

With the aim to reduce computational time and complexity for both training and inference PKM is proposed in [25]. In this context, we eliminate the need to iteratively optimize (4.3) $\forall x$ via $\underline{u}_p \in \{0,1\}$ and $\overline{u}_p \in \{0,1\}, \forall p$ (or equivalently $\{L,R\}$) by replacing them with LPs $\underline{v}_p \in \{0,1\}$ and $\overline{v}_p \in \{0,1\}$. Thus, (4.1) and (4.2) are transformed to:

$$\underline{f}_p^* = \underline{v}_p \overline{f}_p + (1-\underline{v}_p)\underline{f}_p \,,\, \overline{f}_p^* = \overline{v}_p \overline{f}_p + (1-\overline{v}_p)\underline{f}_p \tag{4.7}$$

The reduction in inference complexity with this transformation comes at the cost of an increase in LPs (extra $2P$ LPs), which leads to an increase in learning complexity. Note that, TRS $\tilde{Y}$ and the output y are computed using equations (4.3) and (4.5), respectively, with the substitution of (4.7).

4.2.2 NT CSCMs for DL

In the NT [27], $\tilde{A}_{p,m}$ is reduced to T1-FS $A^*_{p,m}$ as:

$$\mu_{A^*_{p,m}}(x_m) = (\underline{\mu}_{\tilde{A}_{p,m}}(x_m) + \overline{\mu}_{\tilde{A}_{p,m}}(x_m))/2 \tag{4.8}$$

Then, since $\underline{f}_p^* = \overline{f}_p^*$, the output is defined as follows:

$$y(\boldsymbol{x}) = \frac{\sum_{p=1}^{P} [\underline{f}_p(\boldsymbol{x}) y_p + \overline{f}_p(\boldsymbol{x}) y_p]}{\sum_{p=1}^{P} \underline{f}_p(\boldsymbol{x}) + \sum_{p=1}^{P} \overline{f}_p(\boldsymbol{x})} \tag{4.9}$$

and an equivalent TRS can be extracted as follows [5]:

$$\begin{aligned} \underline{y}(\boldsymbol{x}) &= \frac{2\sum_{p=1}^{P} \underline{f}_p(\boldsymbol{x}) \underline{y}_p}{\sum_{p=1}^{P} \underline{f}_p(\boldsymbol{x}) + \sum_{p=1}^{P} \overline{f}_p(\boldsymbol{x})} \\ \overline{y}(\boldsymbol{x}) &= \frac{2\sum_{p=1}^{P} \overline{f}_p(\boldsymbol{x}) \overline{y}_p}{\sum_{p=1}^{P} \underline{f}_p(\boldsymbol{x}) + \sum_{p=1}^{P} \overline{f}_p(\boldsymbol{x})} \end{aligned} \tag{4.10}$$

4.2.2.1 Multi weighted NT (MWNT)

In [25] we provide flexibility by introducing LPs $\gamma_p \in [0,1], \forall p$ within the definition of $A^*_{p,m}$, like in [28], as follows:

$$\mu_{A^*_{p,m}}(x_m) = \gamma_p \underline{\mu}_{\tilde{A}_{p,m}}(x_m) + (1-\gamma_p)\overline{\mu}_{\tilde{A}_{p,m}}(x_m) \tag{4.11}$$

The corresponding defuzzified output is then:

$$y(\boldsymbol{x}) = \frac{\sum_{p=1}^{P}\left[\gamma_p \underline{f}_p(\boldsymbol{x}) y_p + (1-\gamma_p)\overline{f}_p(\boldsymbol{x}) y_p\right]}{\sum_{p=1}^{P}\gamma_p \underline{f}_p(\boldsymbol{x}) + \sum_{p=1}^{P}(1-\gamma_p)\overline{f}_p(\boldsymbol{x})} \tag{4.12}$$

Like in NT, we extract equivalent TRS as follows:

$$\begin{aligned} \underline{y}(\boldsymbol{x}) &= \frac{2\sum_{p=1}^{P}\gamma_p \underline{f}_p(\boldsymbol{x}) \underline{y}_p}{\sum_{p=1}^{P}\gamma_p \underline{f}_p(\boldsymbol{x}) + \sum_{p=1}^{P}(1-\gamma_p)\overline{f}_p(\boldsymbol{x})} \\ \overline{y}(\boldsymbol{x}) &= \frac{2\sum_{p=1}^{P}(1-\gamma_p)\overline{f}_p(\boldsymbol{x}) \overline{y}_p}{\sum_{p=1}^{P}\gamma_p \underline{f}_p(\boldsymbol{x}) + \sum_{p=1}^{P}(1-\gamma_p)\overline{f}_p(\boldsymbol{x})} \end{aligned} \tag{4.13}$$

Thus, the flexibility of NT is increased by the extra P LPs.

4.2.2.2 Weighted NT (WNT)

In this CSCM, we set $\gamma_p = \beta \ \forall p$ in (4.11). Thus, the LP size is reduced from P to 1. $A^*_{p,m}$ is redefined as follows:

$$\mu_{A^*_{p,m}}(x_m) = \beta \underline{\mu}_{\tilde{A}_{p,m}}(x_m) + (1-\beta)\overline{\mu}_{\tilde{A}_{p,m}}(x_m) \tag{4.14}$$

y and $\tilde{Y}$ are computed via (4.12) and (4.13), respectively, with the substitution of (4.14).

4.2.3 BMM CSCM for DL

BMM [29] approximates KM by defining $\underline{f}^*_p$ in (4.1) with $\underline{u}_p = 0$, and $\overline{f}^*_p$ in (4.2) with $\overline{u}_p = 1$, $\forall p$. Thus, (4.3) reduces to:

$$\underline{y}(\boldsymbol{x}) = \frac{\sum_{p=1}^{P}\underline{f}_p(\boldsymbol{x}) y_p}{\sum_{p=1}^{P}\underline{f}_p(\boldsymbol{x})}, \ \overline{y}(\boldsymbol{x}) = \frac{\sum_{p=1}^{P}\overline{f}_p(\boldsymbol{x}) y_p}{\sum_{p=1}^{P}\overline{f}_p(\boldsymbol{x})} \tag{4.15}$$

and the output is defined as:

$$y(\boldsymbol{x}) = \beta \underline{y}(\boldsymbol{x}) + (1-\beta)\overline{y}(\boldsymbol{x}) \tag{4.16}$$

where $\beta \in [0,1]$ is a weighting parameter handled as an LP. If $\beta = 0.5$, the BMM reduces to the Simple Method (SM) [5].

4.3 Addressing the Curse of Dimensionality

The curse of dimensionality is a fundamental problem that appears when fuzzy logic systems are applied to high-dimensional input spaces, which arises in both T1-FLSs and IT2-FLSs as the rapid degeneration of rule firing strengths when the input dimension increases. Similar to representational sparsity and numerical instability observed in deep learning models, increasing the number of inputs causes fuzzy rule activations to collapse toward zero, leading to unreliable inference and ineffective learning. To mitigate this issue, modified firing strength formulations such as High Dimensional TSK (HTSK) [11], Enhanced HTSK (eHTSK), HTSK for IT2-FLSs (HTSK2), and Enhanced HTSK for IT2-FLS (eHTSK2) [25] are examined.

4.3.1 The HTSK

The HTSK approach defines the normalized firing strength $f_p^N(x)$ calculation via a softmax(.) as follows [11]:

$$f_p^N(\boldsymbol{x}) = \frac{\exp(Z_p^{'})}{\sum_{i=1}^{P}\exp(Z_i^{'})} = softmax(Z_p^{'}) \tag{4.17}$$

with

$$Z_p^{'} = \sum_{m=1}^{M} z_{p,m}, \quad z_{p,m} = \frac{1}{M}\frac{(x_m - c_{p,m})^2}{2(\sigma_{p,m})^2} \tag{4.18}$$

4.3.2 The HTSK2

To expand the HTSK to IT2-FLS, we first extract an equivalent (eq) $\mu_{A_{p,m}}(x_m)$ ($\mu^{eq}_{A_{p,m}}(x_m)$). This derivation is used as a basis to define $[\underline{\mu}^{eq}_{\tilde{A}_{p,m}}(x_m), \overline{\mu}^{eq}_{\tilde{A}_{p,m}}(x_m)]$. We start by reformulating the summation in the exponent of $\exp(\sum_{m=1}^{M} z_{p,m})$ in (4.17) as a multiplication of exp(.), which was introduced in [25]:

$$f_p^N(\boldsymbol{x}) \equiv \frac{\prod_{m=1}^{M}\exp(z_{p,m})}{\sum_{i=1}^{P}\left(\prod_{m=1}^{M}\exp(z_{i,m})\right)} \tag{4.19}$$

Then, we can define the following equivalent $f_p(\boldsymbol{x})$ ($f_p^{eq}(\boldsymbol{x})$):

$$f_p^{eq}(\boldsymbol{x}) = \prod_{m=1}^{M}\mu^{eq}_{A_{p,m}}(x_m) \tag{4.20}$$

with an equivalent $\mu_{A_{p,m}}(x_m)$ ($\mu^{eq}_{A_{p,m}}(x_m)$) as follows:

$$\mu^{eq}_{A_{p,m}}(x_m) = \exp\left(-\left(x_m - c_{p,m}\right)^2 / 2(\sigma^{eq}_{p,m})^2\right) \tag{4.21}$$

where $\sigma^{eq}_{p,m} = \sigma_{p,m}\sqrt{M}, \forall p, m$. Thus, we conclude that the deployed softmax(.) within the HTSK is scaling $\sigma_{p,m}$ via M to eliminate $f_p \approx 0$ problem $\forall p$. Now, based on $\mu^{eq}_{A_{p,m}}(x_m)$, we define HTSK2 by defining $\underline{\mu}^{eq}_{\tilde{A}_{p,m}}(x_m)$ and $\overline{\mu}^{eq}_{\tilde{A}_{p,m}}(x_m)$ as:

$$\begin{aligned} \overline{\mu}^{eq}_{\tilde{A}_{p,m}}(x_m) &= \exp\left(-\left(x_m - c_{p,m}\right)^2 / 2(\overline{\sigma}^{eq}_{p,m})^2\right) \\ \underline{\mu}^{eq}_{\tilde{A}_{p,m}}(x_m) &= h_{p,m}\exp\left(-\left(x_m - c_{p,m}\right)^2 / 2(\underline{\sigma}^{eq}_{p,m})^2\right) \end{aligned} \tag{4.22}$$

with $\underline{\sigma}^{eq}_{p,m} = \underline{\sigma}_{p,m}\sqrt{M}$ and $\overline{\sigma}^{eq}_{p,m} = \overline{\sigma}_{p,m}\sqrt{M}$, $\forall p, m$.

4.3.3 The eHTSK2

We also approached the curse of dimensionality problem from a different angle in [25]. Let us first redefine the t-norm operation as:

$$A \cap B = 1 - \min\left((1-A) + (1-B), 1\right) \tag{4.23}$$

Then, the rule firing for a T1-FLS is as follows:

$$f_p(\boldsymbol{x}) = 1 - \min\left(\sum_{m=1}^{M}\left(1 - \mu_{A_{p,m}}(x_m)\right), 1\right) \tag{4.24}$$

Yet, we have transformed the $f_p \approx 0$ problem to $f_p \approx 1$ one when $M \gg 2$ as a result of the min(.). To get rid of it, we redefine (4.24) via the generalized mean approximation as:

$$f_p(\boldsymbol{x}) = 1 - \left[\frac{1}{M}\sum_{m=1}^{M}\left(1 - \mu_{A_{p,m}}(x_m)\right)\right] \tag{4.25}$$

which we named the eHTSK for T1-FLS. The eHTSK method is expanded (4.25) to IT2-FLSs (eHTSK2) as follows:

$$\begin{aligned} \underline{f}_p(\boldsymbol{x}) &= 1 - \left[\frac{1}{M}\sum_{m=1}^{M}\left(1 - \underline{\mu}_{\tilde{A}_{p,m}}(x_m)\right)\right] \\ \overline{f}_p(\boldsymbol{x}) &= 1 - \left[\frac{1}{M}\sum_{m=1}^{M}\left(1 - \overline{\mu}_{\tilde{A}_{p,m}}(x_m)\right)\right] \end{aligned} \tag{4.26}$$

It is worth underlying that the eHTSK/eHTSK2 in (4.25)/(4.26) does not satisfy the properties of the t-norm operator, yet it is differentiable unlike min(.) [30].

4.3.4 HTSK2 vs. eHTSK2: an insight

Here, we provide an insight into the fundamental differences between HTSK2 and eHTSK2. Without loss of generality, we only focus on HTSK and eHTSK. We start by reformulating f_p^{eq} for HTSK given in (4.20) as follows:

$$f_p^{eq}(\boldsymbol{x}) = \prod_{m=1}^{M} (\mu_{A_{p,m}}(x_m))^{1/M} \tag{4.27}$$

and also f_p for eHTSK defined in (4.25) as follows:

$$f_p(\boldsymbol{x}) = \frac{1}{M} \sum_{m=1}^{M} \mu_{A_{p,m}}(x_m) \tag{4.28}$$

Observe that:

- The HTSK performs a linguistic hedge operation on the MFs via M, which is a computationally expensive operation. If $M \gg 2$, occurrence of rules with $f_p^{eq} \approx 0$ is reduced since $\mu_{A_{p,m}}$ is amplified for $\forall p, m$.
- The eHTSK is performing an averaging operation to calculate $f_p(x)$, making it computationally efficient and less sensitive to small variations compared to HTSK. This characteristic helps alleviate the occurrence of $f_p \approx 0$ by smoothing fluctuations in high-dimensional spaces.

5. COMPARATIVE PERFORMANCE ANALYSIS

We present exhaustive comparative results to analyze the learning performances of the proposed DL-based dual-focused IT2-FLS on the White Wine Dataset (WW), Parkinson's Motor UPDRS Dataset (PM), AIDS Dataset (AIDS), Abalone Dataset (ABA), Power Plant Dataset (PP), Boston Housing Dataset (BH), and Concrete Strength Dataset (CS). The investigation encompasses a dual-fold statistical comparison.

(i) An ablation study to examine how the HTSK2 and eHTSK2 handle the curse of dimensionality problem.

(ii) A comparative performance analysis to examine the flexibility and complexity of the proposed CSCMs.

We also present comparative results to analyze the training performance effect of the DL framework integrations, such as Mini-Batch and GPU integration, on the Power Plant Dataset (CCPP), BH, and Energy Efficiency Dataset (ENB). Specifically, a comparative performance analysis to examine the effect of the proposed DL framework on training time.

5.1 Design of Experiment

All datasets are preprocessed by Z-score normalization, and we split 70% of the data into the training set and 30% of the data into the test set. For each dataset, we trained T1-FLSs and IT2-FLSs using identical hyperparameters, including mbs and learning rate, for 100 epochs, except for the Parkinson dataset, where we expanded the training to 1000 epochs. The desired PI coverage was set as 99% for Sec. 5.2, Sec. 5.3, and Sec. 5.4. The experiments were conducted within MATLAB® and repeated with 20 different initial seeds for statistical analysis.

In Sec. 5.2, Sec. 5.3 Sec. 5.4, we formulated eight distinct IT2-FLSs via the presented CSCM. For implementing IT2-FLSs, we used and modified the efficient parallel implementations from Sec. 3.4 and [24]. Each IT2-FLS was configured with the t-norm product operator (PROD), the HTSK2, and the eHTSK2.

All tests were conducted on a PC equipped with an Intel i9-7920 CPU with 12 physical cores and an NVIDIA GTX 1080 Ti GPU.

5.2 Performance Evaluation

For performance evaluation of proposed CSCMs, we evaluated the performances via Root Mean Square Error (RMSE), PI Coverage Probability (PICP), and PI Normalized Averaged Width (PINAW) [31]. We anticipate learning an IT2-FLS that yields a low RMSE (high accuracy) and attains PICP close to 99% with a low PINAW value (i.e., HQ-PI). To analyze the curse of dimensionality, we analyzed the count of "Failed to Train experiments" (F2T), denoting instances where the IT2-FLSs yielded NaNs after the training process. We also recorded the number of "Failed to learn PI models" (FPI), denoting unsuccessful learning instances in which an IT2-FLS produced a PICP $\leq 50\%$ on the testing. The computational load during training is examined by recording the Memory Usage (MU) and total Training Time (TT).

In Table 5.3, we provide the recorded MU and TT values of each model. Table 5.4 provides an RMSE comparison with models from the literature, including T1-FLS and DL variants, to assess the precision performance of the enhanced IT2-FLSs.

In Figure 5.1–5.20, for statistical analysis, we present the notched box and whisker plots obtained from the results of IT2-FLS-HS showing median (central mark), $25^{\text{th}}/75^{\text{th}}$ percentiles (left and right edges of box), i.e. the Inter Quartile Range (IQR), whiskers (line), and outliers (circles).

Remark 3. We excluded the results of the IT2-FLSs that F2T and FPI from all the performance metrics and box plots.

5.3 Ablation Study for Curse of Dimensionality

Let's investigate how the HTSK2 and eHTSK2 address the issue of the curse of dimensionality. Observe that:

- Both HTSK2 and eHTSK2 reduced #F2T in the experiments as is particularly evident in Table 5.1 and Table 5.2 in the PM results.
- Both HTSK2 and eHTSK2 significantly decreased #FPI, resulting in a more robust learning performance. It can be seen that, especially for NT, the #FPI is reduced from 20 to 0 for AIDS, yielding nearly 95% and 96% PICP, respectively, instead of having an unsuccessful PI. Also, #FPI is reduced from 18 to 7 and 0, respectively for PM, yielding nearly 96% and 97% PICP values, respectively.
- While the HTSK2 improved both #F2T and #FPI significantly, eHTSK2 showed superior performance on eliminating #F2T and #FPI, as per their average rankings.
- For equivalent #F2T and #FPI, HTSK2 and eHTSK2 outperform PROD. For example, in the BH case, they exhibit superior RMSE and PICP values.
- When compared to HTSK2, eHTSK2 is not only better at eliminating both #F2T and #FPI but also shows better performance on PINAW as it has a better average ranking. Also, it results in significantly better RMSE and PINAW values on PM and CS as shown in Figure 5.1– 5.4.
- All of the above improvements are also evident in Figure 5.5–5.14.

To sum up, IT2-FLSs, regardless of which CSCM is deployed, defined with both the proposed HTSK2 and eHTSK2, exhibit enhanced capability in addressing the curse of dimensionality. This is evident from the presented overall ranking results. Yet, although both HTSK2 and eHTSK2 exhibit enhanced capability, the effectiveness of eHTSK2 is reflected in the results and gives an edge to eHTSK2 compared to HTSK2.

Table 5.1: Testing performance results of KM, NT, MWNT, SM, and BMM over 20 experiments: curse of dimensionality analysis.

Dataset	Metrics	KM			NT			MWNT			SM			BMM		
$M \times N$		PROD	HTSK2	eHTSK2	PROD	HTSK2	eHTSK2	PROD	HTSK2	eHTSK2	PROD	HTSK2	eHTSK	PROD	HTSK2	eHTSK2
PM 19×5875	RMSE	74.17	63.81	59.95	—	66.01	68.39	—	65.12	61.29	79.68	62.02	61.26	68.16	61.86	62.78
	PICP	98.20	99.33	99.48	—	95.64	97.58	—	90.87	97.71	98.36	98.20	97.68	98.01	97.96	97.55
	PINAW	171.68	149.57	145.11	—	79.30	**80.96**	—	77.68	87.24	102.26	103.84	89.73	97.25	101.74	84.90
	#F2T|#FPI	6|0	1|0	0|0	2|18	0|7	0|0	2|18	0|12	0|0	16|0	0|0	0|0	18|0	1|0	0|0
AIDS 23×2139	RMSE	71.94	72.40	71.70	—	69.96	69.46	—	69.71	69.06	75.30	69.59	68.65	74.77	70.04	69.22
	PICP	98.64	97.62	97.83	—	94.60	96.46	—	96.75	97.67	95.61	97.06	97.31	96.81	97.34	96.99
	PINAW	210.76	194.39	192.86	—	123.13	145.31	—	153.72	176.48	175.25	169.75	178.38	149.45	**155.98**	156.91
	#F2T|#FPI	0|0	0|0	0|0	0|20	0|0	0|0	0|20	0|1	0|0	0|3	0|0	0|0	0|3	0|0	0|0
ABA 8×4177	RMSE	66.41	66.48	**66.25**	67.66	67.69	67.40	68.18	67.08	66.95	67.30	68.60	67.54	66.66	67.40	67.78
	PICP	98.49	98.59	98.54	97.83	98.17	98.06	98.33	98.48	98.38	98.50	98.53	98.46	98.40	98.58	98.42
	PINAW	53.84	52.94	51.70	45.72	48.23	48.25	49.68	51.37	51.42	53.32	52.28	54.45	**45.47**	**45.47**	47.73
	#F2T|#FPI	0|0	0|0	0|0	0|0	0|0	0|0	0|0	0|0	0|0	0|0	0|0	0|0	0|0	0|0	0|0
WW 11×4898	RMSE	82.62	80.61	81.89	81.19	79.52	81.05	80.29	79.37	80.81	81.43	80.46	81.72	81.27	80.63	81.60
	PICP	97.55	98.20	96.95	97.34	98.23	95.91	98.25	98.53	98.26	98.52	98.57	97.59	98.48	98.46	97.48
	PINAW	74.35	70.36	73.43	59.14	62.90	**51.96**	61.31	65.45	64.28	71.37	68.84	62.55	70.88	67.80	61.97
	#F2T|#FPI	3|0	0|0	0|0	0|10	0|2	0|0	0|11	0|1	0|0	3|0	3|0	0|0	1|0	1|0	0|0
PP 4×9568	RMSE	**24.37**	24.52	24.60	24.68	24.66	24.71	24.66	24.68	24.56	24.62	24.62	24.72	24.62	24.61	24.72
	PICP	98.76	98.82	98.83	98.82	98.74	98.69	98.83	**98.92**	98.82	98.66	98.71	98.63	98.71	98.72	98.69
	PINAW	30.81	30.37	30.06	31.83	30.33	30.39	31.91	31.47	29.64	29.81	29.59	**29.33**	29.94	29.73	29.48
	#F2T|#FPI	0|0	0|0	4|0	0|0	0|0	0|0	0|0	0|0	0|0	0|0	0|0	0|0	0|0	0|0	0|0
BH 13×506	RMSE	45.55	42.06	**42.22**	47.38	39.41	41.96	47.79	39.20	40.75	43.35	38.29	42.19	44.73	38.11	41.30
	PICP	95.53	96.78	**97.99**	65.79	90.46	89.90	76.85	93.26	93.75	90.49	92.96	91.68	89.77	93.12	88.85
	PINAW	55.57	49.02	57.00	22.44	29.99	**29.08**	28.20	34.79	36.61	43.39	37.40	39.94	40.51	35.52	34.42
	#F2T|#FPI	0|0	0|0	0|0	0|14	0|0	0|0	0|9	0|0	0|0	0|0	0|0	0|0	0|0	0|0	0|0
CS 8×1030	RMSE	40.32	41.94	38.60	42.21	43.93	35.92	42.50	42.26	34.83	40.84	42.66	35.45	41.49	42.60	35.52
	PICP	98.75	98.96	**99.09**	94.80	97.10	96.02	96.09	96.34	96.70	98.02	98.19	96.05	97.62	98.25	95.42
	PINAW	83.47	88.48	62.94	49.13	**50.31**	34.84	56.55	56.06	37.63	55.65	57.41	42.74	52.40	53.88	40.31
	#F2T|#FPI	5|0	0|0	0|0	0|6	0|0	0|0	0|8	0|0	0|0	1|0	0|0	0|0	0|0	0|0	0|0
Average	RMSE	57.91	55.97	55.03	—	55.88	55.55	—	55.34	**54.03**	58.93	55.17	54.50	57.38	55.03	54.70
	PICP	97.98	98.32	**98.38**	—	96.13	96.08	—	96.16	97.32	96.88	97.46	96.77	96.82	97.49	96.20
	PINAW	97.21	90.73	87.58	—	60.59	60.11	—	67.22	**69.04**	75.86	74.15	71.01	69.41	70.01	65.10
	#F2T+#FPI	2	0.14	0.57	9.99	1.28	0	9.7	2	0	3.27	0.42	0	3.13	0.28	0
Average Rank	RMSE	2.28	2.14	1.57	2.57	1.71	1.71	2.71	1.85	**1.42**	2.14	1.85	1.71	2.28	1.7	2
	PICP	2.57	1.85	1.57	2.57	1.71	1.71	2.85	1.57	1.57	2.14	**1.28**	2.57	2	**1.28**	2.71
	PINAW	2.71	1.85	1.42	2.14	2	1.85	2.14	1.85	2	2.42	1.85	1.71	2.14	2.12	1.42
	#F2T+#FPI	1.57	1.14	1.14	2	1.28	1	2	1.42	1	1.57	1.14	1	1.57	1.28	1
Overall Rank		2.28	1.74	**1.42**	2.32	1.67	1.56	2.43	1.67	1.49	2.06	1.53	1.74	1.99	1.59	1.78

(1) RMSE and PINAW values are scaled by 100. — denotes an absence of a mean value since #F2T + #FPI = 20.

(2) Bold ones indicate the best results in both Table 5.1 and Table 5.2. An IT2-FLS is considered the best if its corresponding PICP falls within the range [97,100] and #F2T+#FPI = 0

Table 5.2: Testing performance results of WKM, PKM, and WNT over 20 experiments: curse of dimensionality analysis.

Datasets $M \times N$	Metrics	WKM PROD	WKM HTSK2	WKM EHTSK	PKM PROD	PKM HTSK2	PKM EHTSK	WNT PROD	WNT HTSK2	WNT EHTSK
PM 19×5875	RMSE	76.19	62.25	**58.55**	76.67	61.01	60.83	—	64.28	63.17
	PICP	97.86	99.09	**99.53**	96.86	98.13	97.85	—	95.65	97.50
	PINAW	165.46	147.81	149.68	117.29	104.11	85.02	—	80.68	89.02
	#F2T|#FPI	9|0	1|0	0|0	0|0	0|0	0|0	1|19	0|10	0|0
AIDS 23×2139	RMSE	69.61	71.37	71.25	72.32	68.99	68.57	—	69.30	**68.45**
	PICP	**98.85**	98.01	97.93	96.31	97.28	97.24	—	96.55	97.13
	PINAW	200.70	188.05	182.73	169.83	171.44	171.60	—	142.86	173.03
	#F2T|#FPI	0|0	0|0	0|0	0|3	0|0	0|0	0|20	0|0	0|0
ABA 8×4177	RMSE	66.64	66.48	66.32	68.18	68.32	67.00	67.89	67.54	66.93
	PICP	98.52	98.56	**98.62**	98.56	98.45	98.49	98.12	98.38	98.41
	PINAW	47.10	46.80	46.22	55.24	54.34	52.09	47.85	49.66	51.63
	#F2T|#FPI	0|0	0|0	0|0	0|0	0|0	0|0	0|0	0|0	0|0
WW 11×4898	RMSE	82.35	80.98	82.07	81.01	**80.10**	81.57	80.76	79.70	81.04
	PICP	97.83	98.32	96.94	98.35	**98.53**	97.92	97.81	98.12	97.58
	PINAW	74.02	70.62	73.87	71.26	67.01	62.43	59.81	64.20	60.66
	#F2T|#FPI	2|0	0|0	1|0	0|0	0|0	0|0	0|10	0|1	0|0
PP 4×9568	RMSE	24.41	24.54	24.59	24.39	24.51	24.55	24.55	24.64	24.64
	PICP	98.76	98.77	98.79	98.77	98.86	98.72	98.80	98.77	98.63
	PINAW	30.90	30.62	30.07	30.42	30.02	**29.33**	31.53	31.39	29.97
	#F2T|#FPI	0|0	0|0	2|0	0|0	0|0	0|0	0|0	0|0	0|0
BH 13×506	RMSE	46.24	41.48	42.24	42.38	38.82	41.35	44.42	38.31	40.86
	PICP	95.13	96.45	97.80	89.80	94.54	91.48	80.56	92.40	90.49
	PINAW	54.73	49.60	55.83	38.92	42.56	32.11	29.15	33.65	31.03
	#F2T|#FPI	0|0	0|0	0|0	0|0	0|0	0|0	0|9	0|0	0|0
CS 8×1030	RMSE	41.97	42.47	**38.25**	40.65	43.10	34.91	41.91	43.52	34.79
	PICP	98.52	98.28	98.67	98.27	98.66	96.21	96.28	97.36	96.05
	PINAW	74.69	78.02	61.56	54.32	58.95	37.43	51.82	51.49	37.83
	#F2T|#FPI	4|0	0|0	0|0	0|0	0|0	0|0	0|6	0|0	0|0
Average	RMSE	58.20	55.65	54.75	57.94	57.9	54.11	—	55.32	54.26
	PICP	97.92	98.21	98.32	96.70	97.77	96.84	—	96.74	96.54
	PINAW	92.51	87.36	85.70	76.75	75.49	67.14	—	64.84	67.59
	#F2T + #FPI	2.14	0.14	0.42	0.42	0	0	9.28	1.57	0
Average Rank	RMSE	2.28	2	1.71	2.28	2	1.71	2.42	1.85	1.57
	PICP	2.42	1.85	1.71	2.28	**1.28**	2.42	2.42	1.57	2
	PINAW	2.71	1.71	1.57	2.42	2.28	**1.28**	2.14	2	1.85
	#F2T + #FPI	1.85	1.14	1.42	1.28	**1**	**1**	2.42	1.28	**1**
Overall Rank		2.31	1.67	1.60	2.06	1.64	1.60	2.35	1.67	1.60

(1) RMSE and PINAW values are scaled by 100. — denotes an absence of a mean value since #F2T + #FPI = 20.
(2) Bold ones indicate the best results in both Table 5.1 and Table 5.2. Best if PICP falls within the range [97,100] and #F2T+#FPI = 0

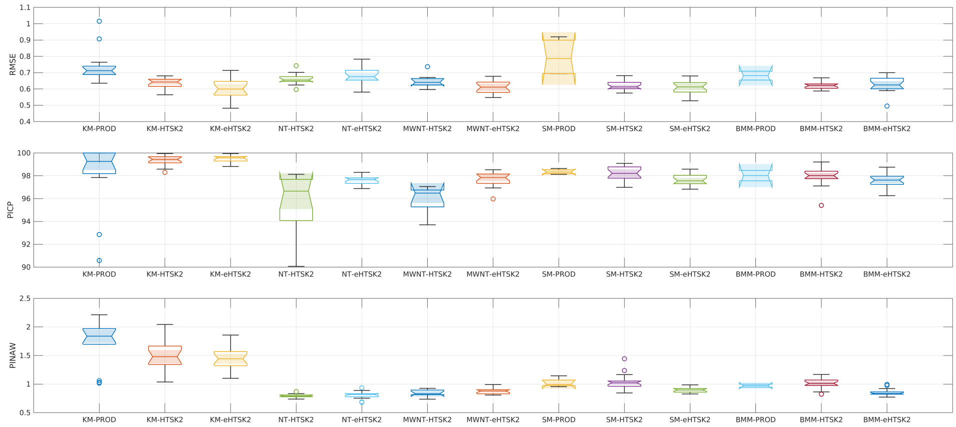


Figure 5.1: Notched box-and-whisker plots for PM Dataset (19×5875): Analyzing the impact of deploying PROD, HTSK2, and eHTSK2 for KM & NT & MWNT & SM & BMM.

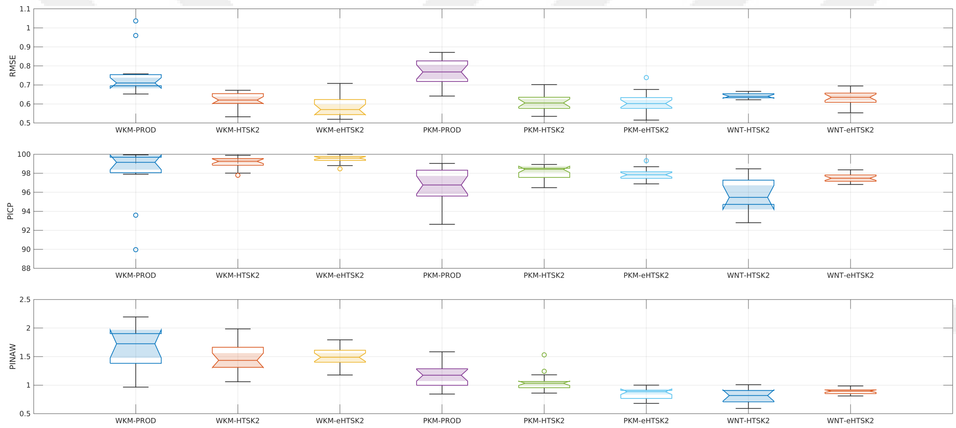


Figure 5.2: Notched box-and-whisker plots for PM Dataset (19×5875): Analyzing the impact of deploying PROD, HTSK2, and eHTSK2 for WKM & WKM & WNT.

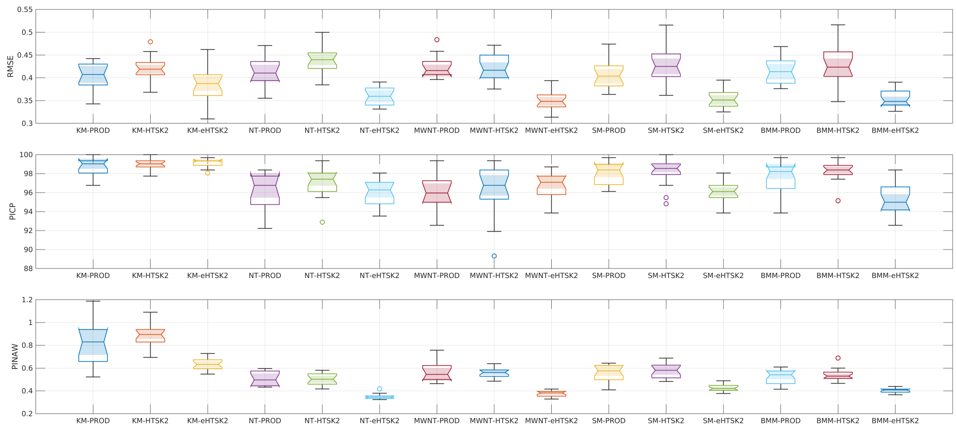


Figure 5.3: Notched box-and-whisker plots for CS Dataset (8×1030): Analyzing the impact of deploying PROD, HTSK2, and eHTSK2 for KM & NT & MWNT & SM & BMM.

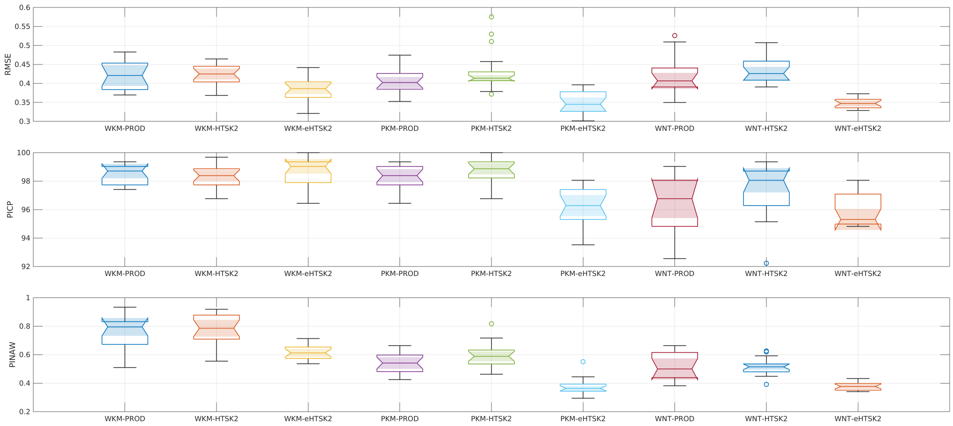


Figure 5.4: Notched box-and-whisker plots for CS Dataset (8×1030): Analyzing the impact of deploying PROD, HTSK2, and eHTSK2 for WKM & WKM & WNT.

The improvements observed in the PM and CS datasets are not unique to those specific cases. Instead, both HTSK2 and eHTSK2 represent a consistent pattern of high performance across all tested scenarios. This demonstrates that the robustness of both HTSK2 and eHTSK2 is further confirmed even when the data characteristics change significantly. To show the full scope of these results, the performance metrics for the remaining datasets are presented in Figure 5.5– 5.14.

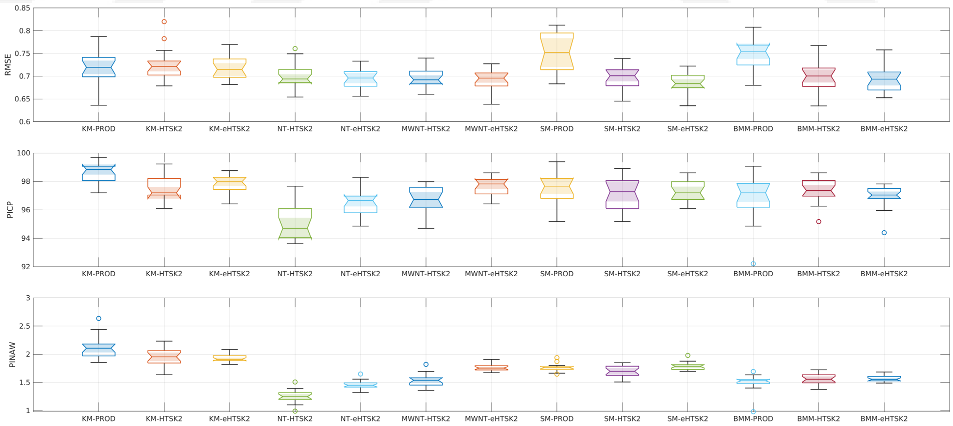


Figure 5.5: Notched box-and-whisker plots for AIDS Dataset (23×2139): Analyzing the impact of deploying PROD, HTSK2, and eHTSK2 for KM & NT & MWNT & SM & BMM.

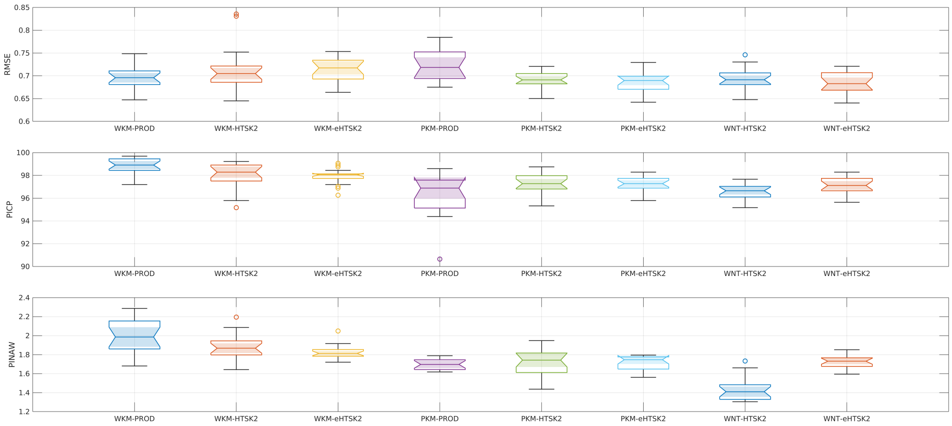


Figure 5.6: Notched box-and-whisker plots for AIDS Dataset (23×2139): Analyzing the impact of deploying PROD, HTSK2, and eHTSK2 for WKM & WKM & WNT.

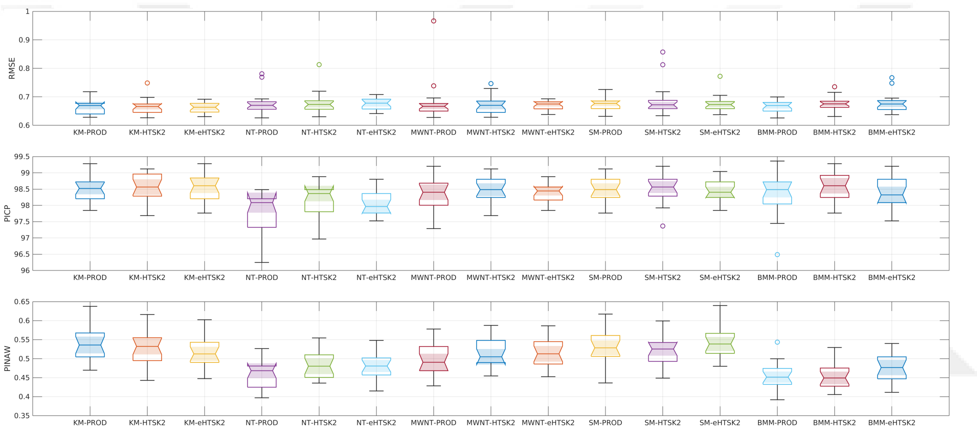


Figure 5.7: Notched box-and-whisker plots for ABA Dataset (8×4177): Analyzing the impact of deploying PROD, HTSK2, and eHTSK2 for KM & NT & MWNT & SM & BMM.

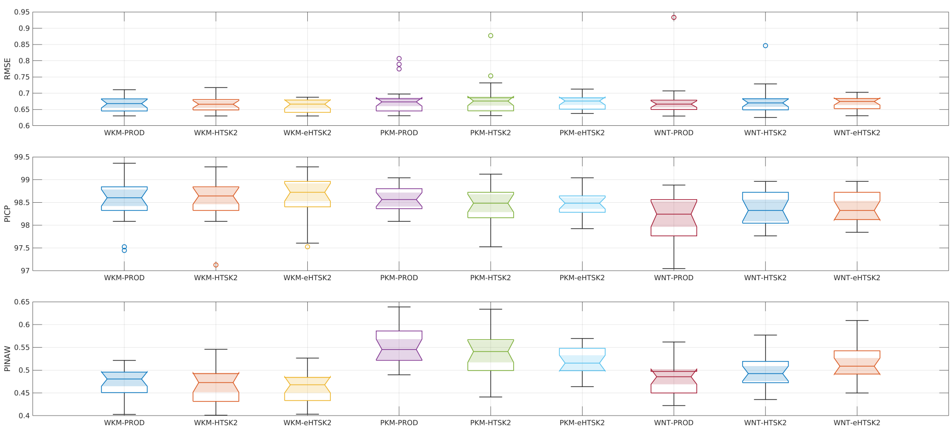


Figure 5.8: Notched box-and-whisker plots for ABA Dataset (8×4177): Analyzing the impact of deploying PROD, HTSK2, and eHTSK2 for WKM & WKM & WNT.

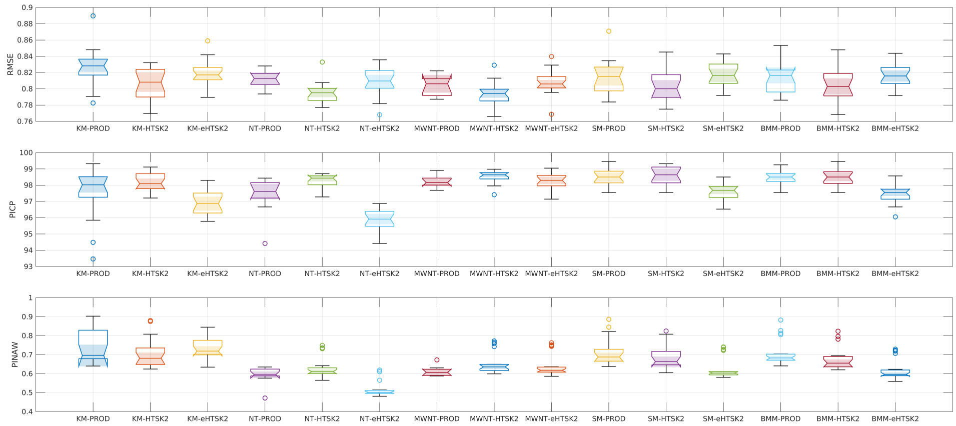


Figure 5.9: Notched box-and-whisker plots for WW Dataset (11×4898): Analyzing the impact of deploying PROD, HTSK2, and eHTSK2 for KM & NT & MWNT & SM & BMM.

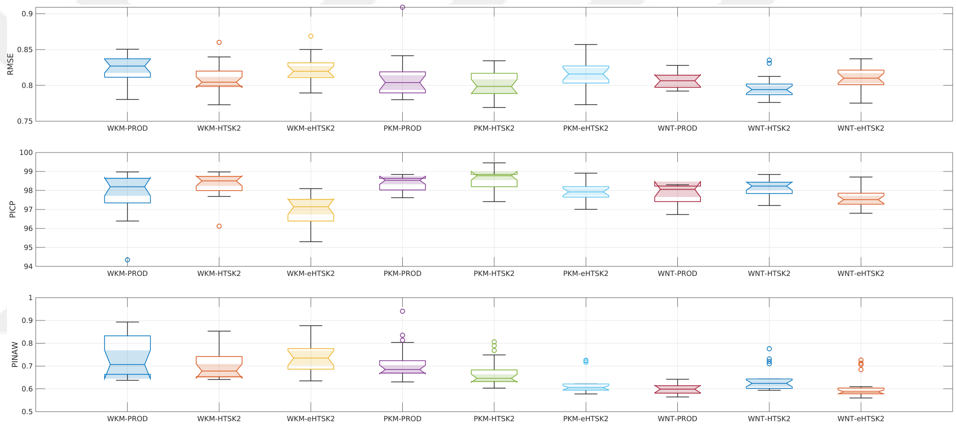


Figure 5.10: Notched box-and-whisker plots for WW Dataset (11×4898): Analyzing the impact of deploying PROD, HTSK2, and eHTSK2 for WKM & WKM & WNT.

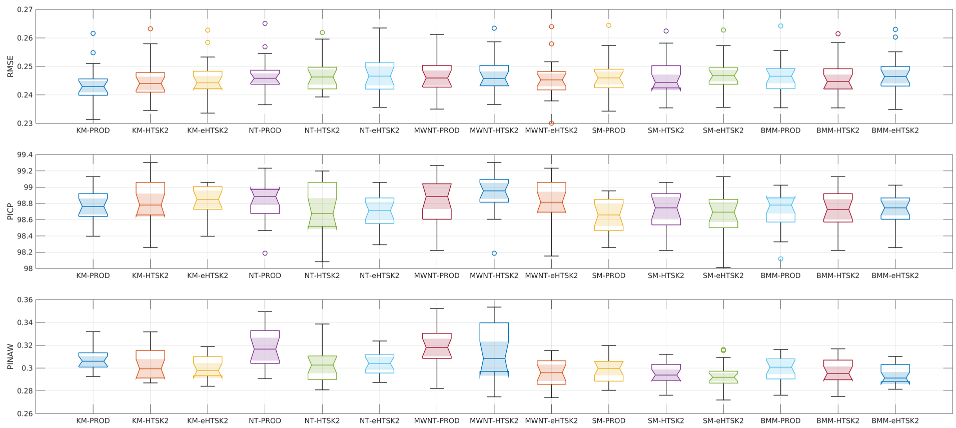


Figure 5.11: Notched box-and-whisker plots for PP Dataset (4×9568): Analyzing the impact of deploying PROD, HTSK2, and eHTSK2 for KM & NT & MWNT & SM & BMM.

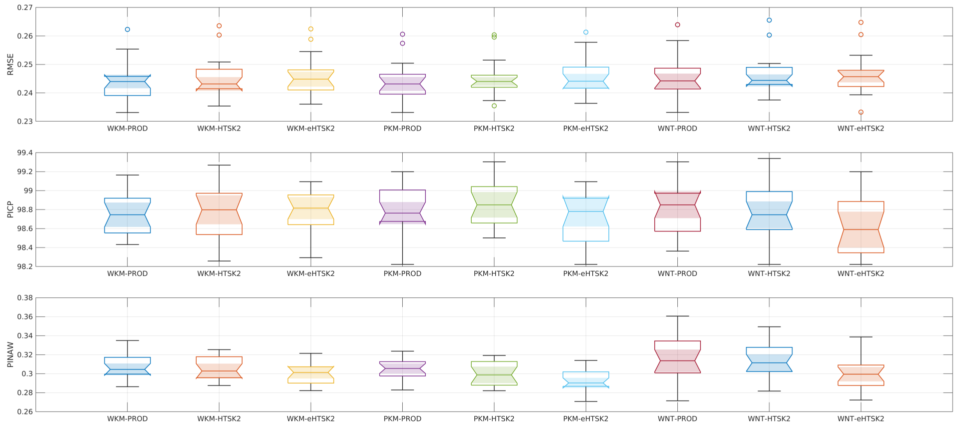


Figure 5.12: Notched box-and-whisker plots for PP Dataset (4×9568): Analyzing the impact of deploying PROD, HTSK2, and eHTSK2 for WKM & WKM & WNT.

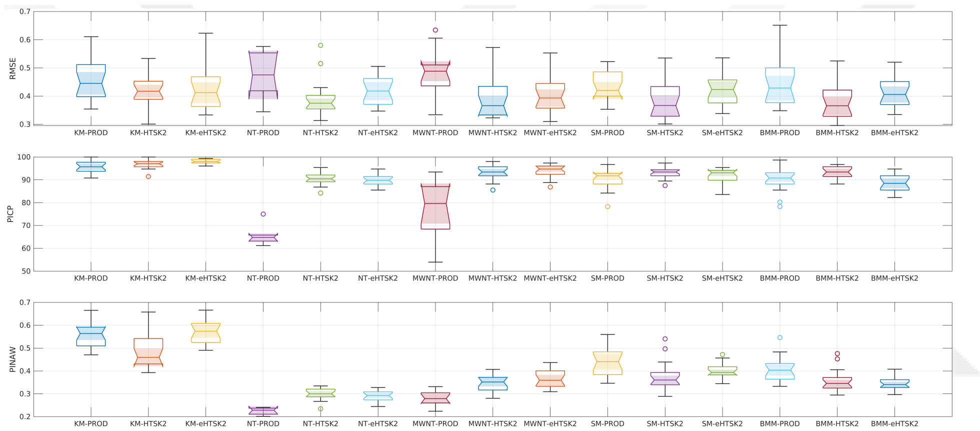


Figure 5.13: Notched box-and-whisker plots for BH Dataset (13×506): Analyzing the impact of deploying PROD, HTSK2, and eHTSK2 for KM & NT & MWNT & SM & BMM.

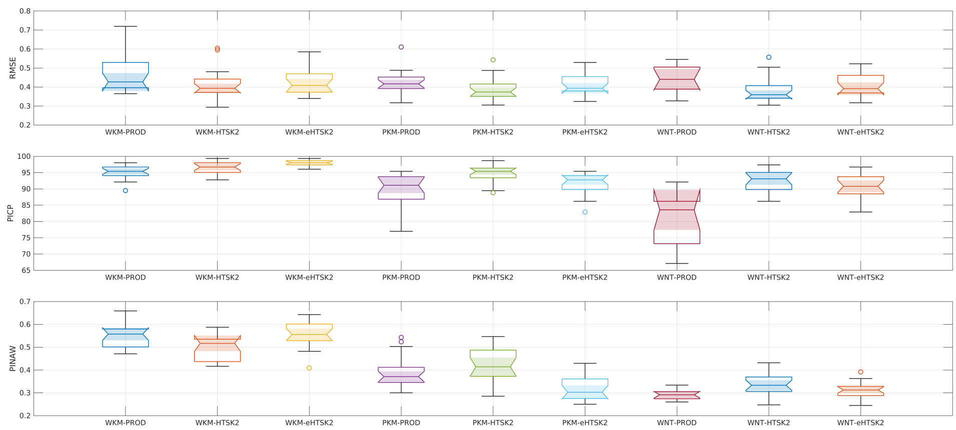


Figure 5.14: Notched box-and-whisker plots for BH Dataset (13×506): Analyzing the impact of deploying PROD, HTSK2, and eHTSK2 for WKM & WKM & WNT.

5.4 Flexibility and Complexity Analysis

Let's explore the performance improvement achieved by deploying the proposed CSCMs with eHTSK2. Observe that:

- WKM and PKM have demonstrated superior performance over their KM version. As can be observed from the AIDS results, WKM has notably smaller PINAW values than KM, and PKM has even smaller PINAW values than KM and WKM. Thus WKM and PKM have better HQ-PI compared to KM. This trend of better performance of PKM on PINAW is even more pronounced in the PM's results. WKM also has a notably better PICP compared to its counterparts as evident from Table 5.5 and Figure 5.15-5.16.
- WNT and MWNT have also demonstrated superior performance over their NT counterpart on RMSE and PICP. If we compare the WNT and the MWNT, MWNT performs better with respect to RMSE and PICP. For instance, WNT achieves better PICP performance compared to NT but WMNT achieves even better RMSE and PICP performance compared to both NT and WNT, such as the WW results given in Table 5.5 and Figure 5.15-5.16.
- All above are also evident in Table 5.6, Table 5.7, and Figure 5.17-5.20.
- In terms of MU, KM and WKM have huge disadvantages compared to other CSCMs. With $P = 15$ rules in the IT2-FLS, they consume over 20 times more memory. As all models are GPU-parallelized, CSCMs demonstrate similar TTs, except when the GPU is saturated, as depicted in Table 5.3. On average, TT exhibits minimal variation among NT, followed by PKM with about a 22% increase, and WKM with approximately a 116% increase.

In the realm of CSCM, all proposed CSCMs have excelled in specific metrics. WKM stands out for its superior PICP, while PKM and WMNT show notable RMSE and PINAW. WKM holds the distinction of having the best RMSE, PICP, and PINAW in seven out of twenty-one cases, marking the highest performance among other CSCMs.

This underscores its robustness in providing reliable HQ-PI. Thus, if one were to recommend a CSCM for high-risk tasks where achieving HQ-PI is crucial, it would be the WKM. Yet, as shown in Table 5.3, WKM comes with a notable computational load. Thus, it is advisable to opt for PKM or MWNT. These alternatives deliver comparable results with lower computational burden, with PKM yielding better total ranking. In Table 5.4, we compared the RMSEs of the PKM with various models that focused solely on the accuracy performance. The comparison includes HTSK (i.e. a T1-FLS with $P = 30$ rules) [10], a Bayesian DL model [32], as well as other ML models [10,32]–[34]. We can conclude that PKM either outperforms or achieves comparable RMSE measures while also being able to generate HQ-PIs.

Table 5.3: Computational load analysis over 20 experiments for $P = 5$ and $P = 15$ rules for CSCMs.

	P	Metrics	KM	WKM	PKM	NT	WNT	MWNT	SM	BMM
PM	5	MU(MB)	78	80	76	**74**	**74**	**74**	76	76
		TT(s)	158	167	171	**136**	149	148	**136**	**136**
	15	MU(MB)	2672	2672	108	**104**	106	**104**	**104**	**104**
		TT(s)	362	375	192	155	168	169	**153**	160
AIDS	5	MU(MB)	**64**	66	**64**	**64**	**64**	**64**	**64**	**64**
		TT(s)	47	48	49	40	48	43	**38**	40
	15	MU(MB)	1022	1024	74	74	**72**	**72**	74	74
		TT(s)	55	58	46	39	43	43	**38**	42
PP	5	MU(MB)	70	72	66	**64**	**64**	**64**	**64**	**64**
		TT(s)	28	28	29	**23**	25	25	**23**	25
	15	MU(MB)	2320	2326	74	**72**	**72**	74	**72**	74
		TT(s)	59	60	29	23	25	26	**23**	24

Bold measures indicate the best results.

Table 5.4: Testing RMSE: PKM-eHTSK2 IT2-FLS vs. various models.

Dataset	PKM-eHTSK2	HTSK [10]	XGBoost [10]	MLP [10]
PP	24.55($\pm$0.66)	22.30($\pm$0.19)	**18.93**($\pm$0.62)	23.15($\pm$0.15)
ABA	67.00($\pm$2.10)	66.46($\pm$0.85)	67.18($\pm$0.85)	**63.59**($\pm$0.83)
PM	**60.83**($\pm$5.32)	82.89($\pm$0.81)	85.61($\pm$0.61)	78.56($\pm$1.83)
Dataset	PKM-eHTSK2	IGS [33]	DtBKS [32]	SDE-MKLS SVM [34]
WW	81.57($\pm$2.03)	84(NP)	**76.8(NP)**	87.8(NP)

(1) RMSE values are scaled by 100. Bold measures indicate the best results.
(2) NP stands for Not Provided.

Table 5.5: Testing performance results over 20 experiments: CSCM analysis with eHTSK2.

Dataset	Metrics	KM	WKM	PKM	NT	WNT	MWNT	SM	BMM
PM	RMSE	59.95(±6.26)	**58.55(±5.56)**	60.83(±5.32)	68.39(±4.59)	63.17(±3.97)	61.29(±3.76)	61.26(±4.56)	62.78(±4.65)
19×5875	PICP	**99.48(±0.31)**	99.53(±0.39)	97.85(±0.57)	97.58(±0.39)	97.50(±0.45)	97.71(±0.61)	97.68(±0.49)	97.55(±0.67)
	PINAW	145.11(±18.09)	149.68(±15.64)	85.02(±9.28)	**80.96(±6.21)**	89.02(±4.70)	87.24(±4.90)	89.73(±4.18)	84.90(±5.74)
AIDS	RMSE	71.70(±2.30)	71.25(±2.52)	68.57(±2.32)	69.46(±2.08)	**68.45(±2.39)**	69.06(±2.38)	68.65(±2.37)	69.22(±2.50)
23×2139	PICP	97.83(±0.67)	**97.93(±0.68)**	97.24(±0.61)	96.46(±0.88)	97.13(±0.77)	97.67(±0.67)	97.31(±0.73)	96.99(±0.81)
	PINAW	192.86(±6.51)	182.73(±7.06)	**171.60(±7.39)**	145.31(±7.51)	173.03(±6.61)	176.48(±6.14)	178.38(±7.02)	156.91(±5.64)
ABA	RMSE	**66.25(±1.89)**	66.32(±1.96)	67.00(±2.10)	67.40(±2.20)	66.93(±2.09)	66.95(±1.67)	67.54(±2.90)	67.78(±3.24)
8×4177	PICP	98.54(±0.44)	**98.62(±0.47)**	98.49(±0.26)	98.06(±0.39)	98.41(±0.34)	98.38(±0.29)	98.46(±0.33)	98.42(±0.43)
	PINAW	51.70(±4.13)	**46.22(±3.51)**	52.09(±3.31)	48.25(±3.59)	51.63(±4.10)	51.42(±3.62)	54.45(±4.27)	47.73(±3.67)
WW	RMSE	81.89(±1.65)	82.07(±2.01)	81.57(±2.03)	81.05(±1.70)	81.04(±1.62)	**80.81(±1.45)**	81.72(±1.34)	81.60(±1.29)
11×4898	PICP	96.95(±0.74)	96.94(±0.76)	97.92(±0.47)	95.91(±0.66)	97.58(±0.54)	**98.26(±0.49)**	97.59(±0.48)	97.48(±0.55)
	PINAW	73.43(±5.51)	73.87(±6.56)	62.43(±5.08)	51.96(±4.36)	**60.66(±5.36)**	64.28(±5.71)	62.55(±5.57)	61.97(±5.31)
PP	RMSE	24.60(±0.72)	24.59(±0.72)	**24.55(±0.66)**	24.71(±0.69)	24.64(±0.71)	24.56(±0.72)	24.72(±0.62)	24.72(±0.68)
4×9568	PICP	**98.83(±0.20)**	98.79(±0.23)	98.72(±0.26)	98.69(±0.20)	98.63(±0.30)	98.82(±0.28)	98.63(±0.31)	98.69(±0.23)
	PINAW	30.06(±1.06)	30.07(±1.15)	**29.33(±1.20)**	30.39(±1.02)	29.97(±1.72)	29.64(±1.21)	**29.33(±1.07)**	29.48(±0.88)
BH	RMSE	**42.22(±7.13)**	42.24(±6.52)	41.35(±6.23)	41.96(±5.23)	40.86(±5.78)	40.75(±6.63)	42.19(±5.21)	41.30(±5.19)
13×506	PICP	**97.99(±1.12)**	97.80(±0.89)	91.48(±3.37)	89.90(±2.35)	90.49(±3.69)	93.75(±2.88)	91.68(±3.55)	88.85(±3.56)
	PINAW	57.00(±5.18)	**55.83(±5.87)**	32.11(±5.86)	29.08(±2.35)	31.03(±3.49)	36.61(±4.26)	39.94(±3.37)	34.42(±2.76)
CS	RMSE	38.60(±3.49)	**38.25(±2.94)**	34.91(±2.98)	35.92(±1.97)	34.79(±1.37)	34.83(±2.34)	35.45(±2.15)	35.52(±2.02)
8×1030	PICP	**99.09(±0.48)**	98.67(±1.04)	96.21(±1.27)	96.02(±1.38)	96.05(±1.21)	96.70(±1.43)	96.05(±1.14)	95.42(±1.57)
	PINAW	62.94(±5.27)	**61.56(±5.16)**	37.43(±5.77)	34.84(±2.33)	37.83(±2.91)	37.63(±2.68)	42.74(±3.08)	40.31(±2.14)
	RMSE	55.03	54.75	54.11	55.55	54.26	**54.03**	54.50	54.70
Average	PICP	**98.38**	98.32	96.84	96.08	96.54	97.32	96.77	96.2
	PINAW	87.58	85.70	67.14	60.11	67.59	**69.04**	71.01	65.10
Average	RMSE	5.28	5.14	3.14	5.71	3	**2.71**	5.28	5.57
Rank	PICP	**2**	2.57	3.71	7	6	3.28	4.57	6.42
	PINAW	7.14	6.42	3.28	**2.28**	3.85	4.42	5.42	3
Overall Rank		4.80	4.71	**3.37**	4.99	4.28	3.47	5.09	4.99

(1) RMSE and PINAW values are scaled by 100.
(2) — denotes an absence of a mean value since #F2T + #FPI = 20.
(3) Bold ones indicate the best results. An IT2-FLS is considered the best if its corresponding PICP falls within the range [97,100].

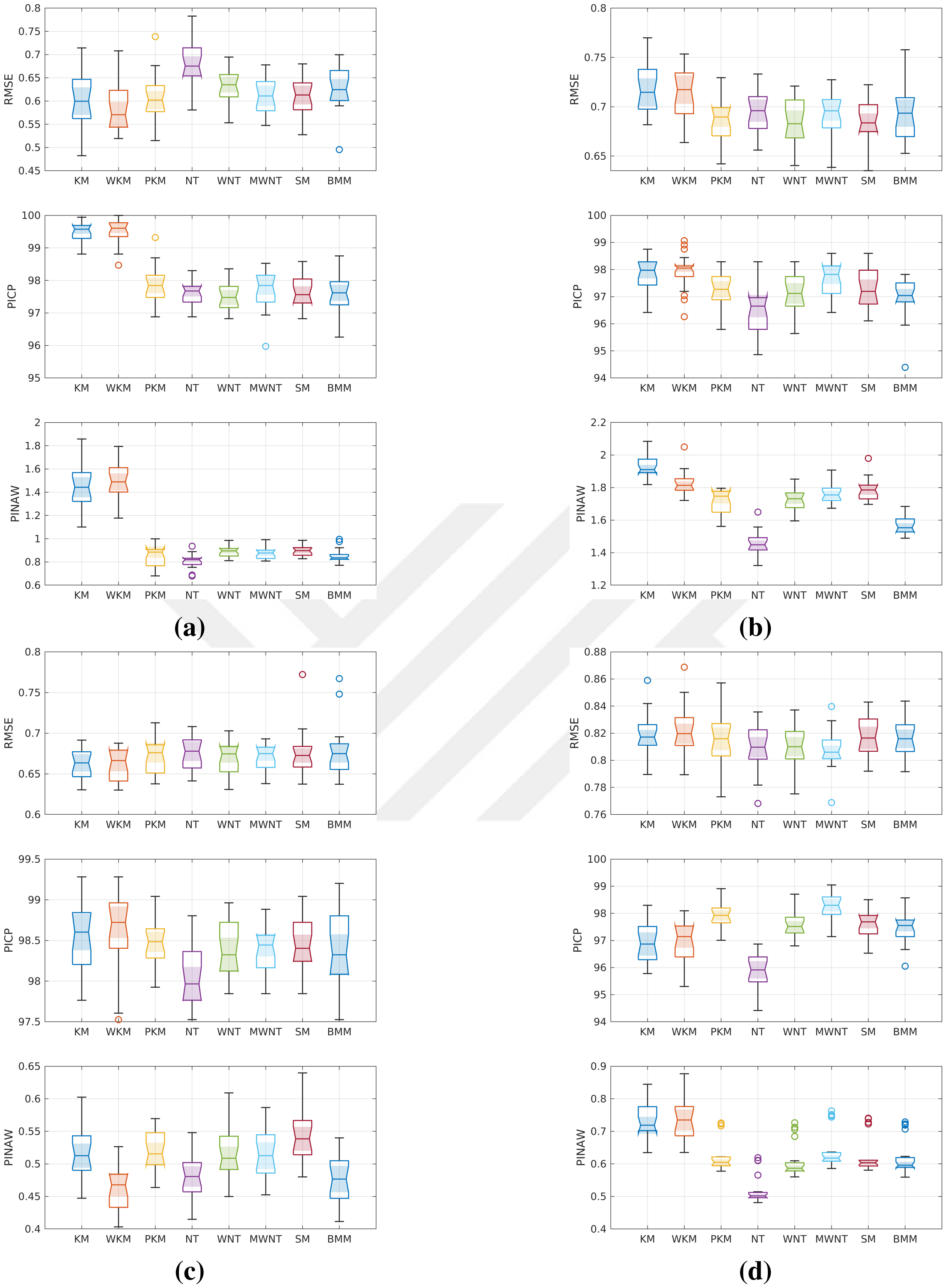


Figure 5.15: Notched box-and-whisker plots of (a)PM, (b)AIDS, (c)ABA, and (d)WW: Analyzing the performance improvements of deploying proposed CSCMs with eHTSK2.

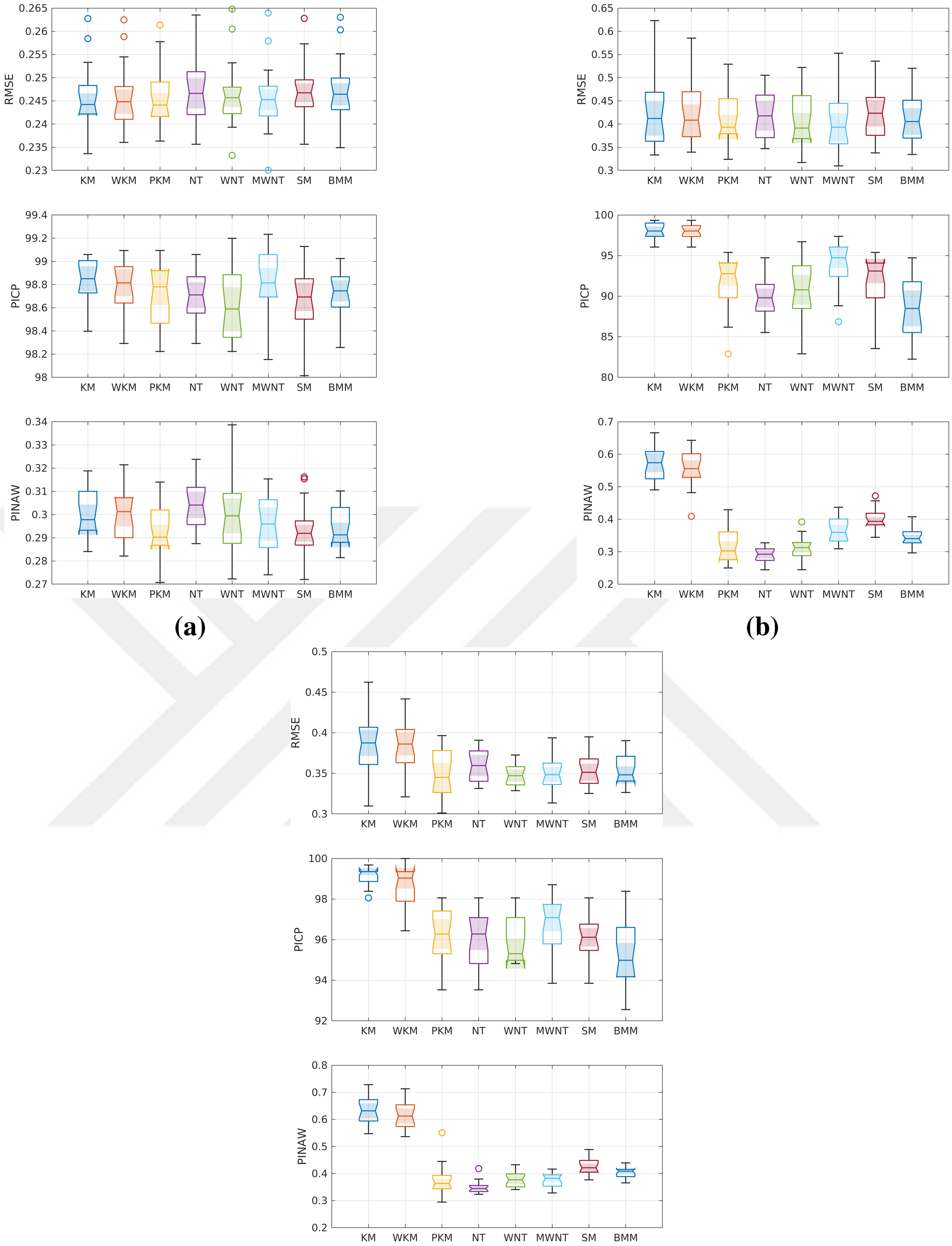


Figure 5.16: Notched box-and-whisker plots of (a)PP, (b)BH, and (c)CS: Analyzing the performance improvements of deploying proposed CSCMs with eHTSK2.

The performance advantages of eHTSK2 and HTSK2 are further supported by the data across different model configurations. Detailed numerical comparisons for HTSK2 and PROD are provided in Table 5.6 and Table 5.7.

Table 5.6: Testing performance results over 20 experiments: CSCM analysis with HTSK2.

Datasets	Metrics	KM	WKM	PKM	NT	WNT	MWNT	SM	BMM
PP	RMSE	63.81(±2.99)	62.25(±3.60)	**61.01**(±4.10)	66.01(±3.55)	64.28(±1.42)	65.12(±4.16)	62.02(±2.89)	61.86(±2.09)
19×5875	PICP	99.33(±0.44)	**99.09**(±0.63)	98.13(±0.64)	95.64(±2.76)	95.65(±1.72)	90.87(±14.23)	98.20(±0.63)	97.96(±0.82)
	PINAW	149.57(±25.68)	147.81(±24.22)	104.11(±14.70)	79.30(±3.63)	80.68(±12.83)	77.68(±19.40)	103.84(±12.73)	**101.74**(±9.19)
AIDS	RMSE	72.40(±3.33)	71.37(±4.66)	**68.99**(±1.77)	69.96(±2.62)	69.30(±2.34)	69.71(±2.45)	69.59(±2.52)	70.04(±3.28)
23×2139	PICP	97.62(±0.92)	**98.01**(±1.16)	97.28(±0.96)	94.60(±2.39)	96.55(±0.71)	96.75(±0.81)	97.06(±1.12)	97.34(±0.85)
	PINAW	194.39(±15.69)	188.05(±12.97)	171.44(±14.47)	123.13(±13.44)	142.86(±12.11)	153.72(±11.59)	169.75(±10.57)	**155.98**(±9.48)
ABA	RMSE	**66.48**(±2.65)	**66.48**(±2.18)	68.32(±5.50)	67.69(±3.87)	67.54(±4.70)	67.08(±3.16)	68.60(±5.53)	67.40(±2.51)
8×4177	PICP	**98.59**(±0.39)	98.56(±0.48)	98.45(±0.41)	98.17(±0.49)	98.38(±0.39)	98.48(±0.40)	98.53(±0.45)	98.58(±0.44)
	PINAW	52.94(±4.29)	46.80(±4.00)	54.34(±5.41)	48.23(±3.33)	49.66(±3.54)	51.37(±3.52)	52.28(±4.11)	**45.47**(±3.59)
WW	RMSE	80.61(±2.05)	80.98(±1.95)	80.10(±1.95)	80.14(±3.01)	79.70(±1.58)	**79.37**(±1.50)	80.46(±2.03)	80.63(±2.11)
11×4898	PICP	98.20(±0.59)	98.32(±0.64)	98.53(±0.50)	95.81(±10.57)	98.12(±0.42)	98.53(±0.39)	**98.57**(±0.58)	98.46(±0.49)
	PINAW	70.36(±7.70)	70.62(±7.27)	67.01(±5.98)	60.40(±12.08)	**64.20**(±5.27)	65.45(±5.77)	68.84(±6.56)	67.80(±5.91)
PP	RMSE	24.52(±0.68)	24.54(±0.70)	**24.51**(±0.65)	24.66(±0.62)	24.64(±0.68)	24.68(±0.65)	24.62(±0.68)	24.61(±0.65)
4×9568	PICP	98.82(±0.30)	98.77(±0.26)	98.86(±0.23)	98.74(±0.31)	98.77(±0.27)	**98.92**(±0.26)	98.71(±0.24)	98.72(±0.25)
	PINAW	30.37(±1.42)	30.62(±1.28)	30.02(±1.30)	30.33(±1.50)	31.39(±2.04)	31.47(±2.54)	**29.59**(±0.94)	29.73(±1.17)
BH	RMSE	42.06(±5.41)	41.48(±7.57)	38.82(±5.65)	39.41(±6.84)	38.31(±6.71)	39.20(±7.37)	38.29(±6.32)	38.11(±6.76)
13×506	PICP	96.78(±1.93)	96.45(±1.82)	94.54(±2.73)	90.46(±2.67)	92.40(±3.25)	93.26(±3.20)	92.96(±2.68)	93.12(±2.70)
	PINAW	49.02(±7.25)	49.60(±5.59)	42.56(±7.79)	29.99(±2.46)	33.65(±4.97)	34.79(±3.66)	37.40(±6.01)	35.52(±4.67)
CS	RMSE	**41.94**(±2.65)	42.47(±2.57)	43.10(±5.10)	43.93(±2.66)	43.52(±3.24)	42.26(±2.95)	42.66(±3.62)	42.60(±3.93)
8×1030	PICP	**98.96**(±0.59)	98.28(±0.87)	98.66(±1.00)	97.10(±1.49)	97.36(±1.77)	96.34(±2.62)	98.19(±1.29)	98.25(±0.97)
	PINAW	88.48(±10.12)	78.02(±10.50)	58.95(±8.56)	**50.31**(±5.02)	51.49(±5.82)	56.06(±4.37)	57.41(±6.23)	53.88(±5.17)
Average	RMSE	55.97	55.65	**54.97**	55.97	55.32	55.34	55.17	55.03
	PICP	**98.32**	98.21	97.77	95.78	96.74	96.16	97.46	97.49
	PINAW	90.73	87.36	75.49	60.24	64.84	67.22	74.15	**70.01**
Average Rank	RMSE	4.42	4.71	**3.28**	6.28	4.42	4.28	4.42	4
	PICP	**2.28**	2.71	3.28	7.42	6.28	4.85	4.57	4.28
	PINAW	7.14	6.42	5.57	**1.85**	3.14	3.85	4.71	3.28
Overall Rank		4.61	4.61	**4.04**	5.18	4.61	4.32	4.56	3.85

(1) RMSE and PINAW values are scaled by 100.
(2) — denotes an absence of a mean value since #F2T + #FPI = 20.
(3) Bold ones indicate the best results. An IT2-FLS is considered the best if its corresponding PICP falls within the range [97,100].

Table 5.7: Testing performance results over 20 experiments: CSCM analysis with PROD.

Datasets	Metrics	KM	WKM	PKM	NT	WNT	MWNT	SM	BMM
PP 19×5875	RMSE	74.17(±10.06)	76.19(±12.14)	76.67(±6.72)	—	—	—	79.68(±11.96)	**68.16**(±3.82)
	PICP	98.20(±2.88)	97.86(±3.18)	96.86(±1.63)	—	—	—	**98.36**(±0.22)	98.01(±0.64)
	PINAW	171.68(±39.58)	165.46(±38.43)	117.29(±21.20)	—	—	—	102.26(±8.34)	**97.25**(±3.58)
AIDS 23×2139	RMSE	71.94(±3.52)	**69.61**(±2.33)	72.32(±3.57)	—	—	—	75.30(±4.52)	74.77(±3.22)
	PICP	98.64(±0.69)	**98.85**(±0.73)	96.31(±2.00)	—	—	—	95.61(±8.05)	96.81(±1.67)
	PINAW	210.76(±19.73)	**200.70**(±19.04)	169.83(±5.46)	—	—	—	175.25(±7.98)	149.45(±15.12)
ABA 8×4177	RMSE	**66.41**(±2.45)	66.64(±2.16)	68.18(±5.05)	67.66(±3.80)	67.89(±6.33)	68.18(±7.12)	67.30(±2.37)	66.66(±1.98)
	PICP	98.49(±0.38)	98.52(±0.47)	**98.56**(±0.29)	97.83(±0.57)	98.12(±0.54)	98.33(±0.52)	98.50(±0.36)	98.40(±0.67)
	PINAW	53.84(±4.41)	47.10(±3.45)	55.24(±4.16)	45.72(±3.70)	47.85(±3.68)	49.68(±4.06)	53.32(±4.74)	**45.47**(±3.57)
WW 11×4898	RMSE	82.62(±2.46)	82.35(±1.90)	**81.01**(±2.95)	82.08(±3.16)	81.77(±3.52)	81.42(±3.77)	81.43(±2.10)	81.27(±1.83)
	PICP	97.55(±1.59)	97.83(±1.15)	98.35(±0.37)	93.32(±13.38)	93.68(±13.74)	93.68(±14.45)	**98.52**(±0.47)	98.48(±0.42)
	PINAW	74.35(±9.12)	74.02(±9.12)	71.26(±7.77)	54.97(±14.51)	55.56(±14.26)	56.48(±15.47)	71.37(±7.29)	**70.88**(±6.88)
PP 4×9568	RMSE	**24.37**(±0.70)	24.41(±0.66)	24.39(±0.69)	24.68(±0.64)	24.55(±0.70)	24.66(±0.65)	24.62(±0.69)	24.62(±0.66)
	PICP	98.76(±0.21)	98.76(±0.24)	98.77(±0.28)	98.82(±0.25)	98.80(±0.27)	**98.83**(±0.30)	98.66(±0.22)	98.71(±0.24)
	PINAW	30.81(±1.07)	30.90(±1.34)	30.42(±1.15)	31.83(±1.70)	31.53(±2.24)	31.91(±2.08)	**29.81**(±1.13)	29.94(±1.12)
BH 13×506	RMSE	45.55(±7.20)	46.24(±8.83)	42.38(±6.31)	47.38(±8.82)	44.42(±6.87)	47.79(±9.25)	43.35(±5.00)	44.73(±7.76)
	PICP	95.53(±2.58)	95.13(±2.24)	89.80(±4.83)	65.79(±4.80)	80.56(±8.48)	76.85(±12.14)	90.49(±4.27)	89.77(±5.57)
	PINAW	55.57(±5.34)	54.73(±5.27)	38.92(±6.97)	22.44(±1.57)	29.15(±2.18)	28.20(±3.19)	43.39(±5.94)	40.51(±5.54)
CS 8×1030	RMSE	**40.32**(±3.24)	41.97(±3.69)	40.65(±3.33)	42.21(±4.15)	41.91(±5.02)	42.50(±2.56)	40.84(±3.29)	41.49(±2.97)
	PICP	**98.75**(±1.03)	98.52(±0.75)	98.27(±0.81)	94.80(±4.68)	96.28(±2.16)	96.09(±1.97)	98.02(±1.16)	97.62(±1.55)
	PINAW	83.47(±19.59)	74.69(±13.46)	54.32(±6.68)	49.13(±8.62)	51.82(±9.31)	56.55(±8.76)	55.65(±7.03)	**52.40**(±6.53)
Average	RMSE	**57.91**	58.20	57.94	—	—	—	58.93	57.38
	PICP	**97.98**	97.92	96.70	—	—	—	96.88	96.82
	PINAW	97.21	**92.51**	76.75	—	—	—	75.86	69.41
Average Rank	RMSE	**2.85**	4	**2.85**	6.42	5	6.42	4	3.28
	PICP	**2.85**	**2.85**	3.42	6.57	5.71	5.57	3.57	4.28
	PINAW	6.42	5.28	4.14	3.42	4.14	5.14	4.14	**2.42**
Overall Rank		4.04	4.04	3.47	5.47	4.95	5.71	3.90	**3.32**

(1) RMSE and PINAW values are scaled by 100.

(2) — denotes an absence of a mean value since $\#F2T + \#FPI = 20$.

(3) Bold ones indicate the best results. An IT2-FLS is considered the best if its corresponding PICP falls within the range [97,100].

In addition to the tabular data, the visual trends across the remaining experimental sets confirm these findings. These box-and-whiskers plots, illustrating the consistency of the proposed methods, are displayed in Figure 5.17–5.20.

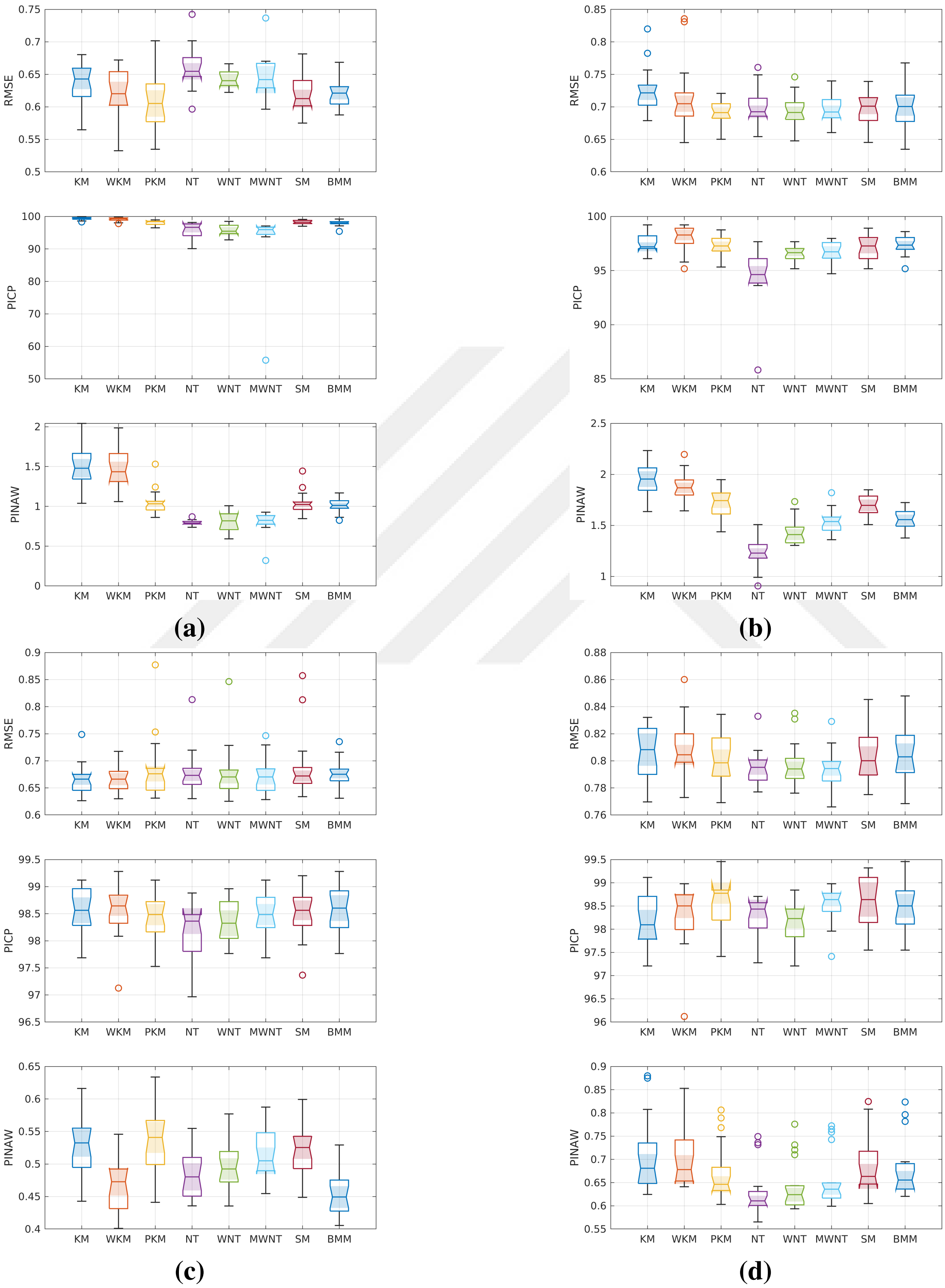
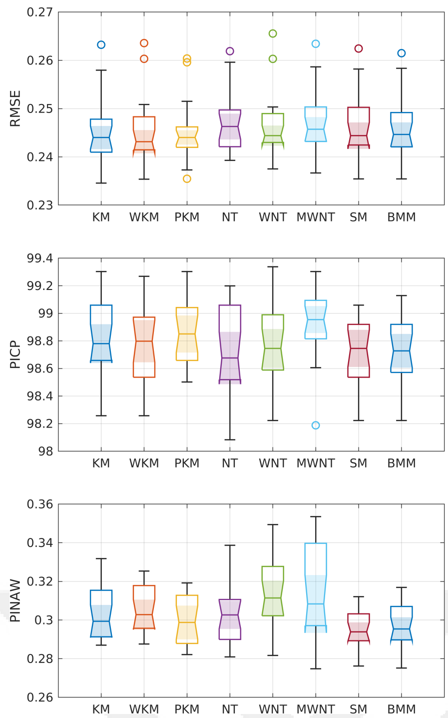
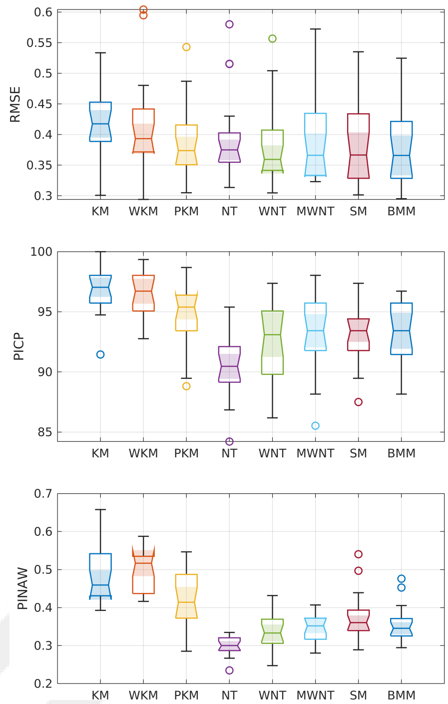


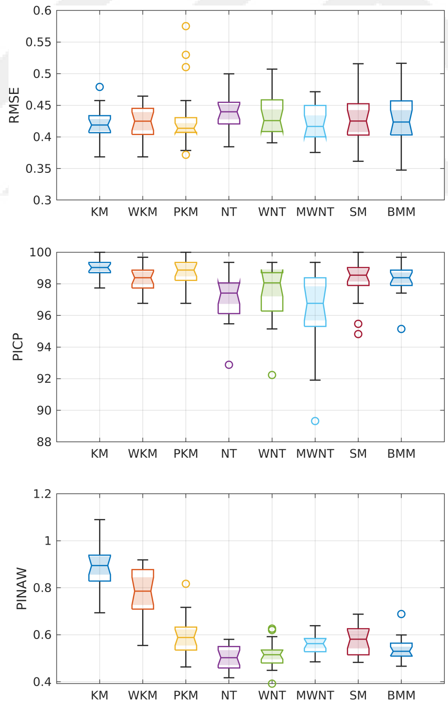
Figure 5.17: Notched box-and-whisker plots of (a)PM, (b)AIDS, (c)ABA, and (c)WW: Analyzing the performance improvements of deploying proposed CSCMs with HTSK2.



(a)



(b)



(c)

Figure 5.18: Notched box-and-whisker plots of (a)PP, (b)BH, and (c)CS: Analyzing the performance improvements of deploying proposed CSCMs with HTSK2.

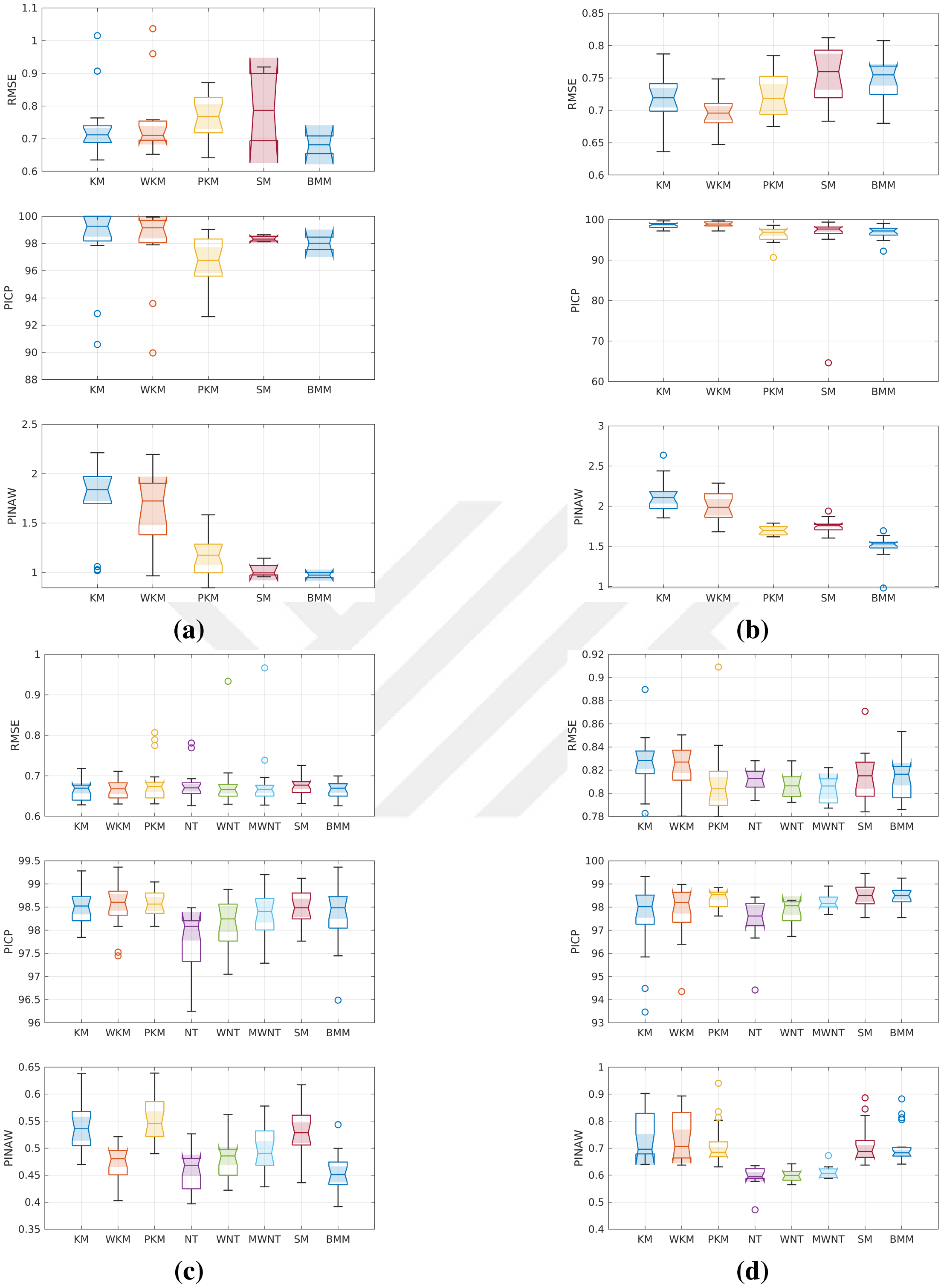


Figure 5.19: Notched box-and-whisker plots of (a)PM, (b)AIDS, (c)ABA, and (d)WW: Analyzing the performance improvements of deploying proposed CSCMs with PROD.

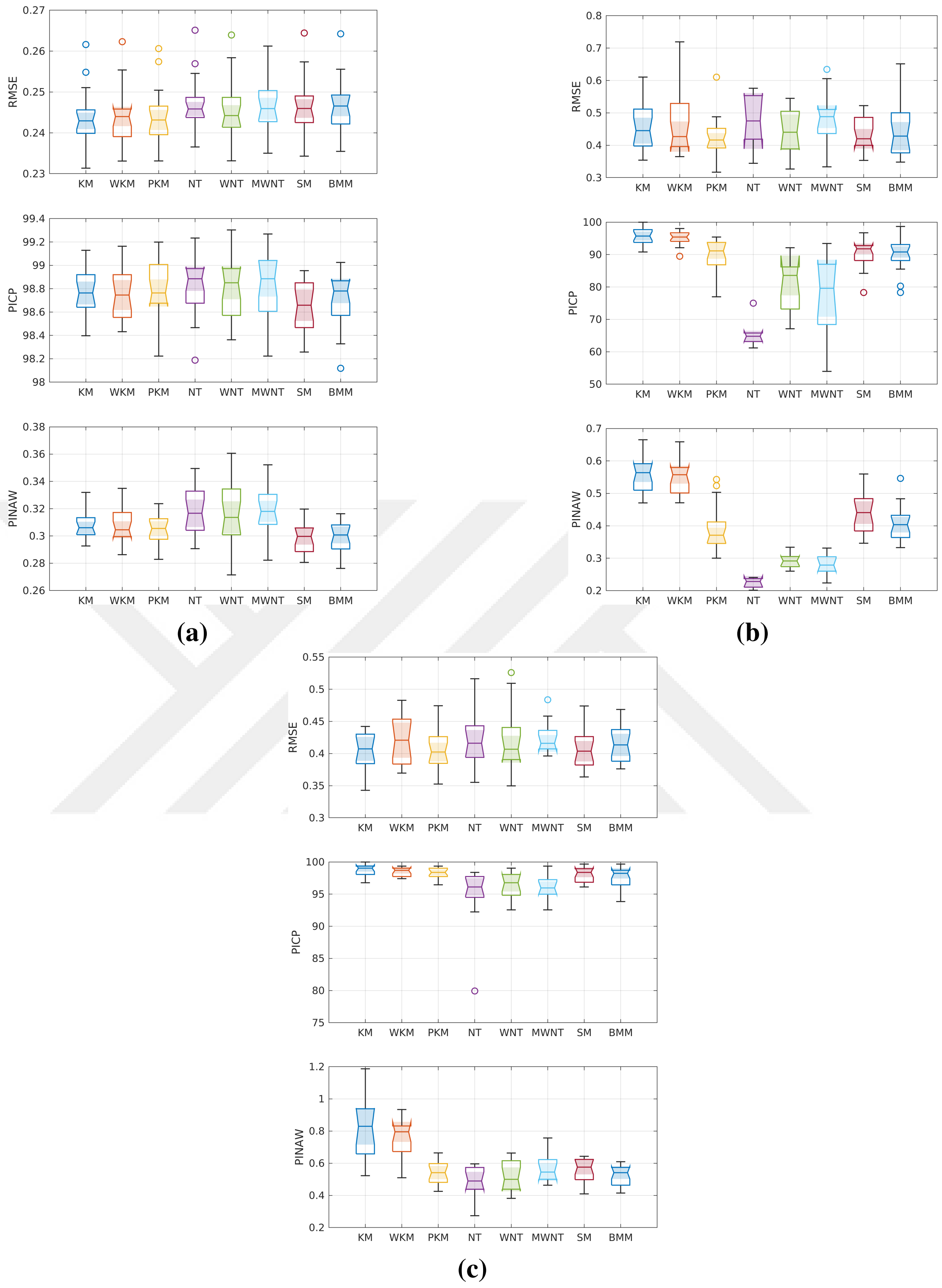


Figure 5.20: Notched box-and-whisker plots of (a)PP, (b)BH, and (c)CS: Analyzing the performance improvements of deploying proposed CSCMs with PROD.

5.5 Performance Analysis of DL Framework Integration

Performance analysis of DL framework integration includes training times and testing RMSE values. As tabulated in Table 5.8, our implementations of T1-FLS and IT2-FLS (abbreviated as IT2-fKM) have accomplished very short training times. Improvement in the training time is visible on fKM compared to the KMA. Our implementation of IT2-FLS was 7218 times faster than the KMA since our implementation computed all of the possible combinations of the KMA in parallel in GPU. Moreover, our implementation is more robust across varying numbers of P and D in terms of training times.

Table 5.8: Performance analysis of DL framework integration.

		Training Time			Test RMSE					
	P	T1	IT2-KM	IT2-fKM	T1		IT2-KM		IT2-fKM	
CCPP	5	11s	18h 16m	18s	0.246		0.242		0.242	
	10	11s	38h 6m	19s	0.241		0.237		0.237	
	15	11s	57h 11m	50s	0.238		0.235		0.235	
BH	5	5s	50m	8s	0.450		0.442		0.442	
	10	5s	1h 38m	8s	0.426		0.428		0.428	
	15	5s	2h 28m	9s	0.416		0.420		0.420	
ENB*	5	7s	3h 2m	12s	0.136	0.203	0.082	0.169	0.082	0.169
	10	7s	6h 50m	13s	0.102	0.173	0.130	0.175	0.130	0.175
	15	7s	9h 57m	45s	0.068	0.153	0.149	0.204	0.149	0.204

*The RMSE values for output-1 and output-2 are provided.

6. CONCLUSIONS

6.1 Conclusion

This thesis presented a unified and computationally efficient framework for learning T1-FLSs and IT2-FLSs within modern DL environments. By reformulating fuzzy system components in a matrix-based and mini-batch-compatible manner, the proposed framework enables integration with automatic differentiation and state of the art DL optimizers, thereby overcoming the scalability and computational limitations of traditional FLS learning approaches.

First, efficient inference implementations were developed to allow FLSs to be trained directly within DL frameworks. Through appropriate parameterization of learning parameters, the originally constrained optimization problems of FLSs were transformed into unconstrained ones, making them suitable to standard DL optimization techniques. Furthermore, mini-batched inference schemes were derived to fully harness parallel computation. For IT2-FLSs, the iterative nature of the KMA was eliminated by introducing a parallelized inference formulation that evaluates all possible type reduction combinations efficiently. Experimental results on widely used benchmark datasets demonstrated that the proposed implementations significantly reduced training time while maintaining, and in some cases improving, prediction accuracy.

In addition to computational efficiency, this thesis addressed fundamental learning challenges of IT2-FLSs, particularly uncertainty quantification and the curse of dimensionality. Novel CSCMs, namely WKM, PKM, WNT, and MWNT, were introduced alongside enhanced high dimensional TSK structures (HTSK2 and eHTSK2). These enhancements improved point-wise prediction accuracy and PI quality while reducing inference complexity. Comprehensive statistical analyses showed that HTSK2 and eHTSK2 effectively mitigated dimensionality issues by substantially reducing the number of firing levels and fuzzy partitions. Among all configurations, eHTSK2 consistently achieved the best overall performance, while PKM emerged as a particularly effective CSCM, offering a strong balance between uncertainty quantification, accuracy, and computational efficiency. Based on these findings, the combination of PKM with eHTSK2 is recommended for efficient and reliable learning of IT2-FLSs.

Overall, this thesis establishes both a practical DL compatible training framework and a set of methodological enhancements that significantly advance the scalability, efficiency, and learning capability of fuzzy logic systems.

6.2 Practical Application of This Study

The outcomes of this study provide practical tools for deploying FLSs in data intensive and real-time applications. The proposed DL-based learning framework enables T1 and IT2-FLSs to be trained on large scale datasets using GPU acceleration and modern optimization techniques, making them suitable for any kind of uncertainty aware systems and applications.

Moreover, the proposed CSCMs and high dimensional TSK structures offer practitioners flexible design choices to balance prediction accuracy, uncertainty quantification, and computational cost. In particular, the recommended PKM & eHTSK2 configuration can be directly applied in scenarios where reliable prediction intervals and efficient inference are critical. The interpretability of fuzzy rules is preserved throughout the framework, allowing domain experts to analyze and validate learned models.

6.3 Future Works

Several directions can be pursued to further extend the contributions of this thesis. First, the proposed learning framework will be extended to alternative DL libraries, such as PyTorch, to improve accessibility and adoption. In addition, integration with TinyML frameworks is planned to enable deployment of trained FLS models on resource-constrained embedded platforms.

From a methodological perspective, future work will focus on extending the learning framework to support diverse shapes of IT2 fuzzy sets and to enable learning of general Type-2 FLSs using zSlices or α-plane representations [1]. Further investigation into adaptive and data-driven structure learning for high dimensional TSK systems is also a promising direction to enhance scalability and robustness in high-dimensional problems.

7. ACKNOWLEDGEMENT

7.1 Funding

The author gratefully acknowledges the financial support provided by MathWorks through a scholarship during the master's studies. The scholarship was associated with a research project funded by MathWorks Research Grant awarded to T. Kumbasar.

REFERENCES

[1] **Mendel, J.M.** (2017). *Uncertain Rule-Based Fuzzy Systems: Introduction and New Directions*, Springer, New York, NY, USA.

[2] **Pekaslan, D.**, **Wagner, C. and Garibaldi, J.M.** (2019). Leveraging IT2 input fuzzy sets in non-singleton fuzzy logic systems to dynamically adapt to varying uncertainty levels, *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pp.1–7.

[3] **Wiktorowicz, K.** (2023). T2RFIS: type-2 regression-based fuzzy inference system, *Neural Comput. Appl.*, *35*(27), 20299–20317.

[4] **Sakalli, A.**, **Kumbasar, T. and Mendel, J.M.** (2021). Towards Systematic Design of General Type-2 Fuzzy Logic Controllers: Analysis, Interpretation, and Tuning, *IEEE Transactions on Fuzzy Systems*, *29*(2), 226–239.

[5] **Beke, A. and Kumbasar, T.** (2023). More Than Accuracy: A Composite Learning Framework for Interval Type-2 Fuzzy Logic Systems, *IEEE Transactions on Fuzzy Systems*, *31*(3), 734–744.

[6] **Pearce, T.**, **Brintrup, A.**, **Zaki, M. and Neely, A.** (2018). High-Quality Prediction Intervals for Deep Learning: A Distribution-Free, Ensembled Approach, *Int. Conf. Mach. Learn.*, volume 80, pp.4075–4084.

[7] **Abdar, M.** *et al.* (2021). A review of uncertainty quantification in deep learning: Techniques, applications and challenges, *Inf. Fusion.*, *76*, 243–297.

[8] **Chung, Y.**, **Neiswanger, W.**, **Char, I. and Schneider, J.** (2021). Beyond pinball loss: Quantile methods for calibrated uncertainty quantification, *Adv. Neural Inf. Process. Syst.*, *34*, 10971–10984.

[9] **Chen, C.**, **Wu, D.**, **Garibaldi, J.M.**, **John, R.I.**, **Twycross, J. and Mendel, J.M.** (2021). A Comprehensive Study of the Efficiency of Type-Reduction Algorithms, *IEEE Transactions on Fuzzy Systems*, *29*(6), 1556–1566.

[10] **Cui, Y.**, **Xu, Y.**, **Peng, R. and Wu, D.** (2023). Layer Normalization for TSK Fuzzy System Optimization in Regression Problems, *IEEE Transactions on Fuzzy Systems*, *31*(1), 254–264.

[11] **Cui, Y.**, **Wu, D. and Xu, Y.** (2021). Curse of Dimensionality for TSK Fuzzy Neural Networks: Explanation and Solutions, *Proc. Int. Jt. Conf. Neural Netw.*, pp.1–8.

[12] **Xue, G.**, **Wang, J.**, **Zhang, K. and Pal, N.R.** (2023). High-Dimensional Fuzzy Inference Systems, *IEEE Trans. Syst. Man Cybern. Syst.*

[13] **Price, S.R.**, **Price, S.R. and Anderson, D.T.** (2019). Introducing Fuzzy Layers for Deep Learning, *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pp.1–6.

[14] **Shihabudheen, K. and Pillai, G.N.** (2018). Recent advances in neuro-fuzzy system: A survey, *Knowl Based Syst.*, *152*, 136–162.

[15] **Zheng, Y.**, **Xu, Z. and Wang, X.** (2021). The fusion of deep learning and fuzzy systems: A state-of-the-art survey, *IEEE Transactions on Fuzzy Systems*, *30*(8), 2783–2799.

[16] **Fumanal-Idocin, J.**, **Andreu-Perez, J.**, **Cordon, O.**, **Hagras, H. and Bustince, H.** (2023). ARTxAI: Explainable Artificial Intelligence Curates Deep Representation Learning for Artistic Images Using Fuzzy Techniques, *IEEE Transactions on Fuzzy Systems*.

[17] **Han, H.**, **Liu, Z.**, **Liu, H.**, **Qiao, J. and Chen, C.P.** (2021). Type-2 fuzzy broad learning system, *IEEE Trans. Cybern.*, *52*(10), 10352–10363.

[18] **de Campos Souza, P.V.** (2020). Fuzzy neural networks and neuro-fuzzy networks: A review the main techniques and applications used in the literature, *Applied soft computing*, *92*, 106275.

[19] **Almaraash, M.**, **Abdulrahim, M. and Hagras, H.** (2023). A Life-Long Learning XAI Metaheuristic-based Type-2 Fuzzy System for Solar Radiation Modelling, *IEEE Transactions on Fuzzy Systems*.

[20] **Aliev, R.A.** *et al.* (2011). Type-2 fuzzy neural networks with fuzzy clustering and differential evolution optimization, *Inf. Sci.*, *181*(9), 1591–1608.

[21] **Das, R.**, **Sen, S. and Maulik, U.** (2020). A survey on fuzzy deep neural networks, *ACM Computing Surveys*, *53*(3), 1–25.

[22] **Zhang, B.**, **Wang, J.**, **Zhang, C.**, **Yang, J.**, **Kumbasar, T. and Wu, W.** (2024). Zero-order fuzzy neural network with adaptive fuzzy partition and its applications on high-dimensional problems, *Neurocomputing*, *569*, 127118.

[23] **Karnik, N.N. and Mendel, J.M.** (2001). Centroid of a type-2 fuzzy set, *Information Sciences*, *132*(1), 195–220.

[24] **Koklu, A.**, **Guven, Y. and Kumbasar, T.** (2024). Efficient Learning of Fuzzy Logic Systems for Large-Scale Data Using Deep Learning, *International Conference on Intelligent and Fuzzy Systems*, p.406–413.

[25] **Koklu, A.**, **Guven, Y. and Kumbasar, T.** (2025). Odyssey of Interval Type-2 Fuzzy Logic Systems: Learning Strategies for Uncertainty Quantification, *IEEE Transactions on Fuzzy Systems*, *33*(1), 468–478.

[26] **Kumbasar, T.** (2017). Revisiting Karnik–Mendel Algorithms in the framework of Linear Fractional Programming, *Int. J. Approx. Reason.*, *82*, 1–21.

[27] **Nie, M. and Tan, W.W.** (2008). Towards an efficient type-reduction method for interval type-2 fuzzy logic systems, *IEEE Int. Conf. Fuzzy Syst.*, pp.1425–1432.

[28] **Runkler, T.A.**, **Chen, C. and John, R.** (2018). Type reduction operators for interval type–2 defuzzification, *Inf. Sci.*, *467*, 464–476.

[29] **Begian, M.B.**, **Melek, W.W. and Mendel, J.M.** (2008). Parametric design of stable type-2 TSK fuzzy systems, *Annual Meeting of the North American Fuzzy Information Processing Society*, pp.1–6.

[30] **van Krieken, E.**, **Acar, E. and van Harmelen, F.** (2022). Analyzing Differentiable Fuzzy Logic Operators, *Artificial Intelligence*, *302*, 103602.

[31] **Quan, H.**, **Srinivasan, D. and Khosravi, A.** (2014). Short-Term Load and Wind Power Forecasting Using Neural Network-Based Prediction Intervals, *IEEE Trans. Neur. Net. Learn. Syst.*, *25*(2), 303–315.

[32] **Partaourides, H. and Chatzis, S.** (2018). Deep learning with t-exponential Bayesian kitchen sinks, *Expert Syst. Appl.*, *98*, 84–92.

[33] **Wu, D.**, **Lin, C.T. and Huang, J.** (2019). Active learning for regression using greedy sampling, *Information Sciences*, *474*, 90–105.

[34] **Liu, C.**, **Tang, L. and Liu, J.** (2019). Least squares support vector machine with self-organizing multiple kernel learning and sparsity, *Neurocomputing*, *331*, 493–504.

APPENDICES

APPENDIX A: Code Repositories

Appendix A: Code Repositories

The source code and supplementary materials developed in this thesis are publicly available at the following repositories:

- Repository 1: Efficient Learning of Fuzzy Logic Systems for Large Scale Data Using Deep Learning
- Repository 2: Odyssey of Interval Type-2 Fuzzy Logic Systems Learning Strategies for Uncertainty Quantification

CURRICULUM VITAE

Name SURNAME: Ata KÖKLÜ

EDUCATION:

- **B.Sc.:** 2023, Istanbul Technical University, Faculty of Electrical and Electronics Engineering, Department of Control and Automation Engineering

PROFESSIONAL EXPERIENCE AND REWARDS:

- 2025-Present Control System Design Engineer at ASELSAN A.Ş.
- 2024 Recipient of the IEEE Computational Intelligence Society Travel Grant for paper presentation at the IEEE World Congress on Computational Intelligence (WCCI 2024).
- 2023–2024 MathWorks Researcher at the Artificial Intelligence and Intelligent Systems Laboratory (AI2S), Istanbul Technical University, within the “Developing High-Performance Deep Neural Networks with Fuzzy Layer” project funded by MATLAB, MathWorks.
- 2021–2023 Student Researcher at the Artificial Intelligence and Intelligent Systems Laboratory (AI2S), Istanbul Technical University.
- 2020–2021 Embedded Software Engineer at Facilis Racing Technologies, Borusan Automotive Motorsports.

PUBLICATIONS, PRESENTATIONS AND PATENTS ON THE THESIS:

- **Koklu, A.**, Guven, Y., Kumbasar, T. (2024). Efficient Learning of Fuzzy Logic Systems for Large-Scale Data Using Deep Learning. *International Conference on Intelligent and Fuzzy Systems*, July 16-18, 2024 Çanakkale, Turkey.
- **Koklu, A.**, Guven, Y., Kumbasar, T. (2024). Enhancing Interval Type-2 Fuzzy Logic Systems: Learning for Precision and Prediction Intervals. *IEEE International Conference on Fuzzy Systems, IEEE World Congress on Computational Intelligence*, June 30- July 5, 2024 Yokohama, Japan.
- Guven, Y., **Koklu, A.**, Kumbasar, T. (2024). Zadeh’s Type-2 Fuzzy Logic Systems: Precision and High-Quality Prediction Intervals. *IEEE International Conference on Fuzzy Systems, IEEE World Congress on Computational Intelligence*, June 30- July 5, 2024 Yokohama, Japan.

- **Koklu, A.**, Guven, Y., Kumbasar, T. (2024). Odyssey of Interval Type-2 Fuzzy Logic Systems: Learning Strategies for Uncertainty Quantification, *IEEE Transactions on Fuzzy Systems,* 33(1), 468-478.

- Guven, Y., **Koklu, A.**, Kumbasar, T. (2024). Exploring Zadeh's General Type-2 Fuzzy Logic Systems for Uncertainty Quantification, *IEEE Transactions on Fuzzy Systems,* 33(1), 314-324.

OTHER PUBLICATIONS, PRESENTATIONS AND PATENTS:

- Sertbas, A.E., **Koklu, A.**, Kumbasar, T. (2024). System Identification and State Estimation with Deep Learning Supported State-Space Equations. *25. Otomatik Kontrol Ulusal Konferansı*, September 12-14, 2024 Konya, Turkey.

- Koklu, K., **Koklu, A.** (2019). On the Sums of Some Special Number Sequence. *Zeugma II. International Multidisciplinary Studies Congress*, January 18-20, 2019 Gaziantep, Turkey.