RESEARCH PORTFOLIO

Selected Projects in Applied and Computational Mathematics, Fluid Dynamics, and Artificial Intelligence

ABSTRACT

This portfolio presents a collection of my research works at the intersection of Applied Mathematics, Computational Fluid Dynamics, and Artificial Intelligence. Each project reflects a progressive exploration of mathematical modeling, neural-network-based simulations, and data-driven analysis for solving real-world scientific and healthcare problems.

Jalal Uddin

Visiting Lecturer — Dep. Mathematical Sciences
BUITEMS University
M.S. Mathematics — COMSATS University Islamabad
Email: jalalkhilji954@gmail.com
GitHub: https://github.com/jalal954
Kaggle: https://www.kaggle.com/jalal954

<u>M.S. Thesis Project</u> — Survival Data Analysis of Childhood Disease with Competing Risk Models

Objective:

To analyze the survival patterns and risk factors of **congenital hypothyroidism (CH)** in children and adults using statistical survival models that account for **competing risks** and multiple outcomes.

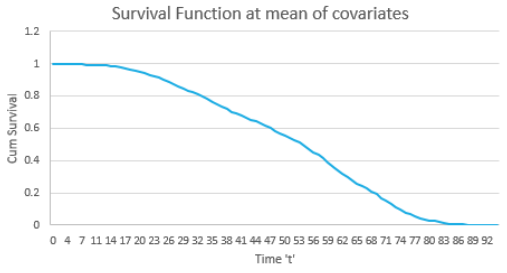
Methodology:

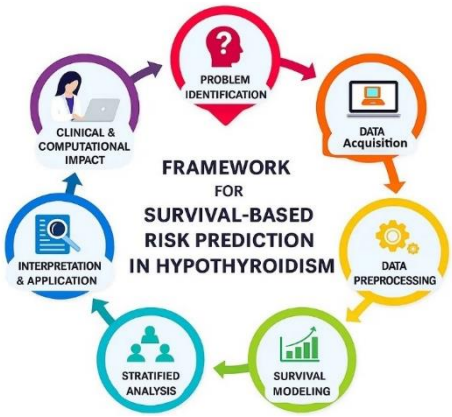
A dataset of 3,772 thyroid patient record**s** (UCI Machine Learning Repository) was analyzed. Data preprocessing involved missing-value handling, variable encoding, and exploratory data analysis (EDA) using SPSS and Python. Models applied included Cox Proportional Hazards Regression**,** Competing Risk Regression**,** and Logistic Regression to estimate survival time and hazard ratios under multiple risk conditions.

Key Findings:

- Age, TSH, and thyroxine therapy were identified as key predictors of hypothyroidism survival outcomes.
- Female patients exhibited a higher incidence rate (~66%).
- The model achieved strong predictive accuracy ($R^2 \approx 0.84$) with validated proportional hazard assumptions.
- Results highlighted the importance of thyroxine treatment in improving survival among affected individuals.

Skills & Tools: SPSS · Python · Cox Regression · Survival & Hazard Modeling · Data Visualization.





<u>Project – 1.</u> Optimizing heat transfer in Blasius Flow of Tetra Hybrid Nanofluids Using LMA Neural Networks

(AI-Based Optimization of Blasius Flow of Tetra-Hybrid Nanofluids Using LMA–Neural Networks)

Objective:

To enhance the heat-transfer efficiency of Blasius flow using a tetra-hybrid nanofluid model and optimize its behavior through a Levenberg–Marquardt Neural Network (LMA–NN).

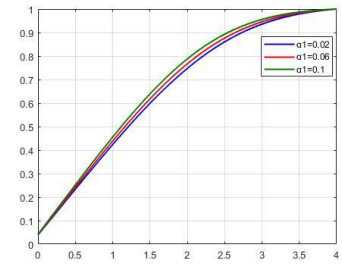
Methodology:

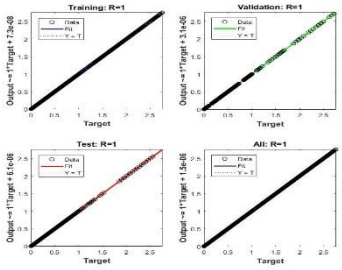
Developed a Tetra-Hybrid Blasius Model (THBM) for laminar boundary-layer flow and generated reference data using the Additive Runge–Kutta (ARK) numerical scheme. The LMA–NN, implemented in MATLAB, was trained (70 % train, 15 % validation, 15 % test) to predict velocity and temperature profiles under varying parameters such as nanoparticle concentration, slip number, density, thermal radiation (Rd), and Eckert number (Ec).

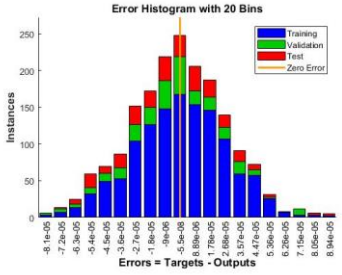
Key Findings:

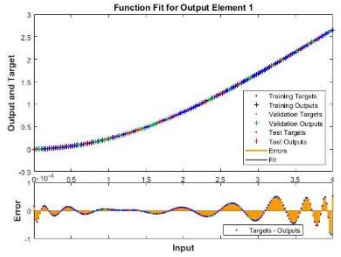
The network achieved very low MSE (10^{-9}–10^{-10}) with strong regression ($R \approx 1$). Results show that higher nanoparticle concentration, slip number, and nanofluid density increase flow velocity, while higher radiation and slip factors reduce temperature. Demonstrated that LMA–NNs can accurately emulate and optimize complex nanofluid systems**,** offering a reliable tool for energy and heat-transfer applications.

Skills & Tools: MATLAB · Neural Networks · CFD · Heat-Transfer Modeling · Data-Driven Simulation.









<u>Project - 2.</u> AI-Driven Analysis of Unsteady Casson Fluid Flow and Heat Transfer Over a Stretching Surface Using Levenberg-Marquardt Neural Networks

(AI-Driven Simulation of Unsteady Casson Fluid Flow and Heat Transfer Using LMA–Neural Networks)

Objective:

To analyze and predict unsteady Casson fluid flow and heat transfer over a stretching surface using an AI-based neural network solver**,** improving accuracy for complex non-Newtonian models.

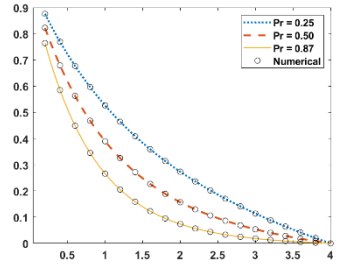
Methodology:

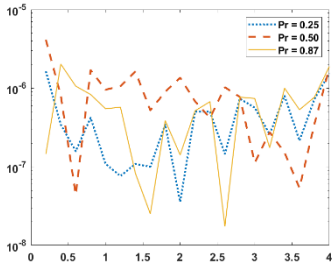
Applied similarity transformations to convert the governing PDEs to ODEs, then generated reference data with the Additive Runge–Kutta (ARK) method. A Levenberg–Marquardt Neural Network (LMA–NN) was trained in MATLAB (72 % training, 14 % validation, 14 % testing) for varying Casson (β), Prandtl (Pr), and unsteadiness (A) parameters to predict velocity and temperature profiles.

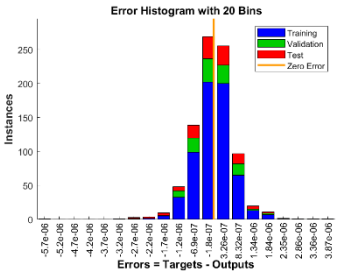
Key Findings:

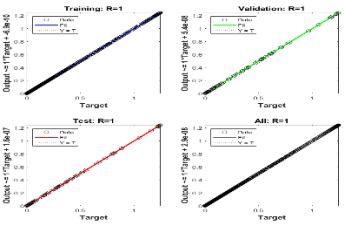
The LMA–NN achieved very low MSE (10^{-10}–10^{-13}) and near-perfect regression ($R \approx 1$). Results show that higher β increases temperature while reducing velocity, and higher Pr and A lower temperature fields. The approach proved highly accurate and computationally efficient, demonstrating AI's capability to solve nonlinear fluid-dynamics problems relevant to manufacturing and energy systems.

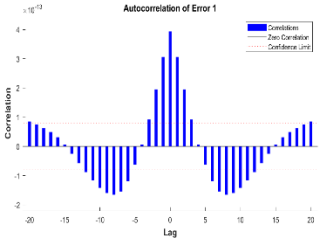
Skills & Tools: MATLAB · Neural Networks · Casson Fluid Modeling · CFD · Non-Newtonian Flow Simulation











<u>Project – 3.</u> Towards Artificial Neural Networking of Newtonian Fluid Flow with External Constraints

(Intelligent Surrogate for Newtonian Fluid Flow and Heat–Mass Transfer in Porous Media)

Objective:

To develop a fast and accurate neural-network surrogate for Newtonian boundary-layer flow over a porous flat plate with suction, including coupled heat and mass transfer effects.

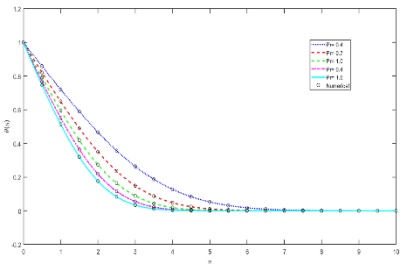
Methodology:

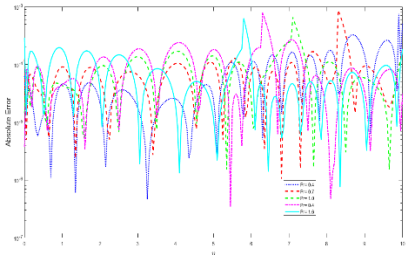
The governing boundary-layer equations were reduced to ODEs by similarity transformation and solved numerically using the Additive Runge–Kutta (ARK) method to create training data. A Levenberg–Marquardt Neural Network (LMA–NN) was implemented in MATLAB and trained on these data (70 % train / 15 % validation / 15 % test) to predict velocity, temperature, and concentration fields for varying Prandtl (Pr), Schmidt (Sc), Soret (Sr), and porous-resistance parameters.

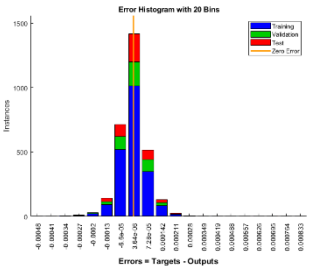
Key Findings:

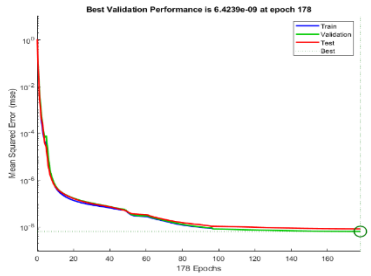
The LMA–NN achieved $R \approx 1$ and MSE $\leq 10^{-9}$, matching the numerical solver point-for-point. Results confirmed that higher porous resistance parameter thins the momentum layer, larger Prandtle Pr increases wall heat transfer, higher Schmidt number Sc suppresses diffusion, and stronger Soret number Sr thickens the concentration boundary layer. The model proved to be a stable, computationally efficient surrogate for parametric CFD analysis.

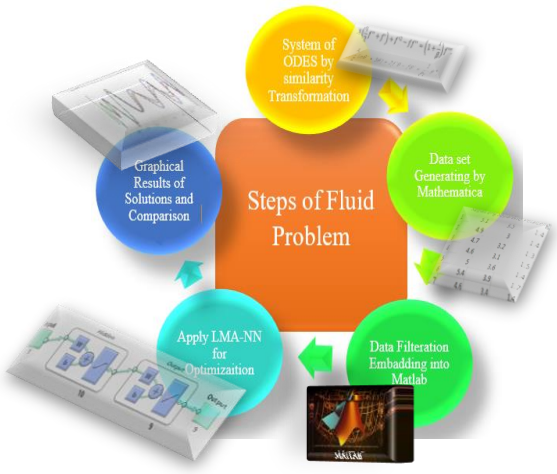
Skills & Tools: MATLAB · LMA–NN · CFD · Heat/Mass Transfer · Porous-Media Modeling











<u>Project – 4. (Ongoing)</u> An Explainable Hybrid Deep Learning Model Combining GAN and TabNet for Imbalanced Thyroid Disease Diagnosis

Objective:

To design an explainable hybrid deep learning model that integrates Generative Adversarial Networks (GANs) for data balancing and TabNet for transparent classification of thyroid disorders.

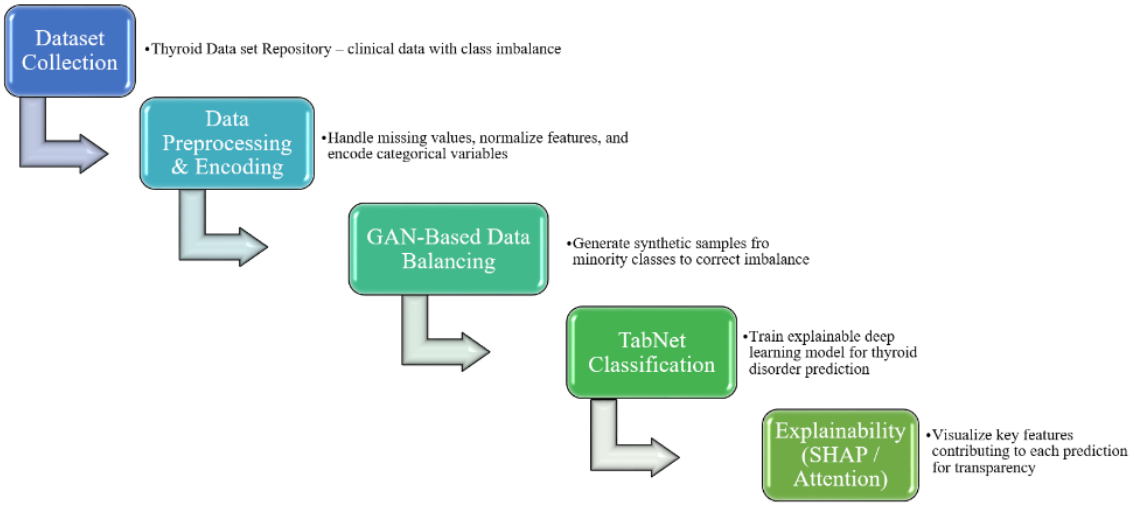
Methodology:

The project focuses on handling imbalanced medical datasets from a publicly available thyroid disease dataset by using GANs to generate synthetic minority samples and employing TabNet for interpretable diagnosis through feature-level attention. The framework aims to combine data augmentation, predictive modeling, and explain-ability in a single pipeline. Evaluation will involve standard metrics such as accuracy, recall, and F1-score, alongside explain-ability tools like SHAP for visual feature interpretation. The stepwise process of the proposed Hybrid GAN–TabNet framework is shown in the diagram below.

Expected Outcomes:

The hybrid model is expected to improve prediction performance for underrepresented thyroid disease cases while maintaining model transparency and fairness. This approach seeks to demonstrate how interpretable AI can support reliable decision-making in medical diagnostics.

Skills & Tools: Python · PyTorch · GANs · TabNet · SHAP · Explainable AI · Imbalanced Data Processing



<u>Project – 5.</u> Logistic Regression–Based Classification of Thyroid Disease

Objective:

⛓ **GitHub Repo:** https://github.com/Jalal954/Thyroid-Disease-Classification-using-Logistic-Regression
⛓ **Kaggle NB:** https://www.kaggle.com/code/jalal954/thyroid-classification-logistic-regression

To develop a baseline machine learning model using Logistic Regression for the classification of thyroid disease cases into *hypothyroid (1)* versus *negative (0)* classes. The aim was to build an interpretable diagnostic framework and evaluate model performance on a real-world medical dataset.

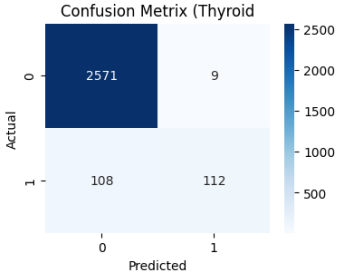
Methodology:

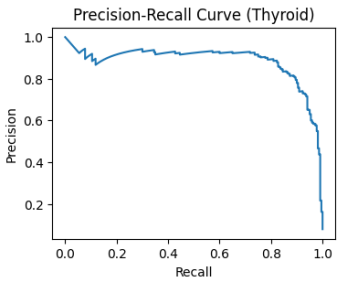
A dataset of 3,772 patient records (UCI Machine Learning Repository) was analyzed using Python. A custom $wrangle()$ function was designed to perform one-step data preprocessing, including replacement of "?" with NaN, conversion of numerical columns, and imputation of missing values with median (for numeric) and mode (for categorical) methods. To address class imbalance, all hypothyroid-related subclasses (hypothyroid, primary hypothyroid, compensated hypothyroid, and secondary hypothyroid) were merged into a single positive class (1). The cleaned data were split (80/20) for training and testing, and a Logistic Regression model was fitted and evaluated using **accuracy, precision, recall, F1-score, and ROC–AUC** metrics.

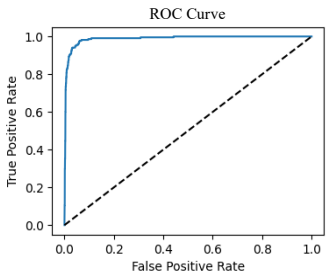
Key Findings:

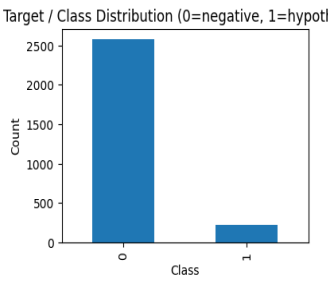
- The model achieved $Accuracy = 95.82\%$**,** $Precision = 92.56\%$**,** $Recall = 50.91\%$**,** $F1 = 0.66$**,** and $ROC-AUC = 0.9886$, indicating excellent class separation capability.
- The model performed exceptionally well for the majority class but missed some minority cases (recall ≈ 51%), a common issue in imbalanced medical data.
- Visual diagnostics (Confusion Matrix, ROC, PR curves) confirmed model reliability, and feature coefficients identified key predictors such as TSH**,** T4U, and clinical flags (on thyroxine, thyroid surgery).
- The study establishes a strong baseline for future work integrating class balancing (SMOTE/GAN) and explainable AI (TabNet) approaches.

Skills & Tools: Python · Pandas · NumPy · Scikit-learn · Matplotlib · Seaborn · Kaggle · GitHub · Data Wrangling · Imbalanced Classification.









Project – 6. Attention-Based Multimodal Emotion Analysis Using Deep Learning

(Technical Assessment Project for TUAI MSCA Doctoral Candidate Position, 2025)
(Explainable Emotion Recognition on CMU-MOSI Dataset)

Objective:

GitHub Repo: https://github.com/Jalal954/Multimodal-Emotion-Analysis-Attention

To design and implement an explainable multimodal deep learning model that combines textual and visual features through an attention-based fusion mechanism for emotion recognition. This work was developed during the Marie Skłodowska-Curie Actions (MSCA) TUAI Doctoral Candidate Technical Assessment, where the author advanced to the final evaluation stage, reflecting strong research and analytical potential.

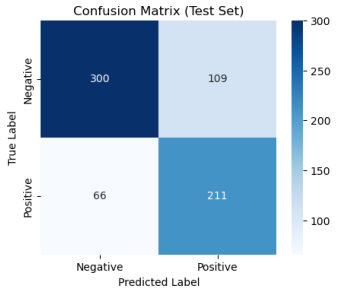
Methodology:

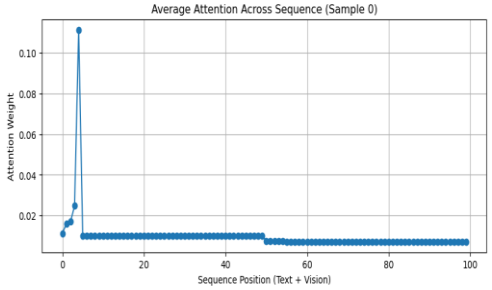
The study employed the CMU-MOSI dataset via the CMU-MultimodalSDK for emotion classification tasks. Textual and visual features were projected into a shared latent space and fused through a multi-head attention mechanism implemented in PyTorch. The framework emphasizes both performance and interpretability, incorporating attention heatmaps and feature-importance visualization to explain decision pathways. Model performance was evaluated using accuracy and F1-score metrics.

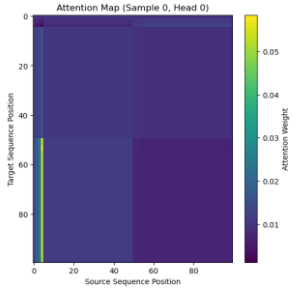
Key Findings:

- The attention-based fusion approach effectively captured inter-modality relationships for improved emotion recognition.
- Visual analysis of attention weights demonstrated high model interpretability, identifying key linguistic and facial cues influencing classification.
- Further hyperparameter tuning (learning rate, attention depth, and embedding dimensions) could yield higher accuracy and model robustness.

Skills & Tools: Python · PyTorch · NumPy · Scikit-learn · Matplotlib · Seaborn · Explainable AI · Multimodal Fusion







Project – 7. Multi-Class Image Classification using CNN (Keras + TensorFlow)

Objective:

⚯ **GitHub Repo:** https://github.com/Jalal954/CNN-MultiClass-Image-Classification
⚯ **Kaggle NB:** https://www.kaggle.com/code/jalal954/multi-class-image-classification-using-cnn-keras

To develop a convolutional neural network (CNN) capable of accurately classifying natural images into seven distinct real-world categories bike, cars, cats, dogs, flowers, horses, and human. The project aimed to construct a baseline deep-learning framework for computer-vision tasks and analyze model behavior using modern evaluation metrics.

Methodology:

A total of seven labeled image folders were used from the Kaggle dataset. All images were resized to 180 × 180 pixels and normalized to the [0, 1] range. The dataset was divided into 80 % training and 20 % validation subsets using a fixed random seed for reproducibility.
A sequential CNN architecture was implemented in TensorFlow Keras, consisting of three convolution pooling blocks followed by fully connected dense layers and a softmax output layer. The model was trained using the Adam optimizer and Sparse Categorical Cross-Entropy loss for 27 epochs with a batch size of 32.
Performance was assessed using training/validation accuracy, classification report (precision, recall, F1-score), and a confusion-matrix visualization.

Key Findings:

- The CNN achieved **98.3 %** training accuracy and **~70 %** validation accuracy, showing effective learning and moderate generalization on unseen images.
- The macro F1-score **(0.67)** demonstrated balanced performance across all seven categories.
- The model performed strongly for structured and high-contrast classes (bike, cars, flowers), while mild confusion was observed between visually similar classes (cat's vs dogs).
- Training and validation loss curves confirmed stable convergence without severe overfitting.
- Results validate that a properly tuned CNN can deliver high-accuracy multi-class classification on moderately sized, unaugmented image datasets and can serve as a foundation for later transfer-learning or data-augmentation improvements.

Skills & Tools: Python · TensorFlow · Keras · Convolutional Neural Networks · Image Classification · Matplotlib · NumPy · Scikit-learn · GitHub · Kaggle.

Layer (type)	Output Shape	Param #
rescaling (Rescaling)	(None, 180, 180, 3)	0
conv2d (Conv2D)	(None, 178, 178, 32)	896
max_pooling2d (MaxPooling2D)	(None, 89, 89, 32)	0
conv2d_1 (Conv2D)	(None, 87, 87, 64)	18,496
max_pooling2d_1 (MaxPooling2D)	(None, 43, 43, 64)	0
conv2d_2 (Conv2D)	(None, 41, 41, 128)	73,856
max_pooling2d_2 (MaxPooling2D)	(None, 20, 20, 128)	0
flatten (Flatten)	(None, 51200)	0
dense (Dense)	(None, 128)	6,553,728
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 7)	903